Deteksi & Prediksi Kesehatan ayam dengan *YOLOv5 - K-Nearest Neighbors*

Tugas Akhir

Oleh:
Marshanda Simanjuntak (4212201035)
Aldi Ananda Pakpahan (4212201061)

Program Studi Teknik Mekatronika
Jurusan Teknik Elektro
Politeknik Negeri Batam
2026

Pernyataan Keaslian Tugas Akhir

Pernyataan Keaslian Tugas Akhir

Saya yang bertandatangan dibawah ini menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya yang berjudul : "Deteksi & Prediksi Kesehatan ayam dengan YOLO - K-Nearest Neighbors" adalah **hasil karya sendiri**, **diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan**, dan **bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.** Semua referensi yang dikutip atau dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan saya ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Batam, 19 Januari 2026

Marshanda Simanjuntak
NIM: 4212201035

Aldi Ananda Pakpahan
NIM: 4212201061

Lembar Pengesahan

Lembar Pengesahan

Tugas Akhir disusun untuk memenuhi salah satu syarat memperoleh gelar
Sarjana Terapan Teknik (S.Tr.T)
di
Politeknik Negeri Batam

Oleh:
Marshanda Simanjuntak (4212201035)
Aldi Ananda Pakpahan (4212201061)

Tanggal Sidang: 19 Januari 2026

Disetujui oleh :

1. Abdullah Sani S.ST, M.Sc
   NIK: 113119

1. M. Naufal Airlangga Diputra
   S.Pd., M.P.H
   NIK: 122281

2. Indra Hardian Mulyadi S.T.,
   M.Eng., Ph.D
   NIK: 117179

Deteksi & Prediksi Kesehatan Ayam dengan *YOLOv5- K-Nearest Neighbors*

Abstrak

Kesehatan ayam petelur merupakan faktor penting yang berpengaruh langsung terhadap produktivitas dan kualitas hasil peternakan. Namun, proses pemantauan kesehatan ayam secara manual masih memiliki keterbatasan, seperti ketergantungan pada pengamatan subjektif dan membutuhkan waktu yang relatif lama. Oleh karena itu, penelitian ini bertujuan untuk merancang dan mengimplementasikan sistem deteksi dan prediksi kesehatan ayam petelur berbasis *computer vision* menggunakan metode *YOLOv5* dan *K-Nearest Neighbors (KNN)*. Metode *YOLOv5* digunakan untuk mendeteksi objek ayam secara otomatis dari citra digital, sedangkan algoritma *KNN* dimanfaatkan untuk mengklasifikasikan kondisi kesehatan ayam berdasarkan fitur visual yang diperoleh. Proses penelitian meliputi pengumpulan dataset citra ayam, tahap *preprocessing*, pelatihan model, serta pengujian sistem secara menyeluruh. Hasil pengujian menunjukkan bahwa model *YOLOv5* mampu mendeteksi objek ayam dengan tingkat akurasi sebesar **92,4%**, sedangkan algoritma *KNN* menghasilkan nilai akurasi klasifikasi sebesar **90,1%**. Selain itu, nilai *precision* dan *recall* yang diperoleh masing-masing sebesar **89,6%** dan **91,2%**, yang menandakan performa klasifikasi yang cukup seimbang. Pengujian validitas sistem menunjukkan bahwa hasil prediksi yang dihasilkan memiliki tingkat kesesuaian yang tinggi terhadap kondisi aktual ayam, dengan nilai validitas sebesar **0,89**. Sementara itu, uji reliabilitas sistem menghasilkan nilai sebesar **0,91**, yang mengindikasikan bahwa sistem mampu memberikan hasil yang konsisten pada pengujian berulang. Berdasarkan hasil tersebut, dapat disimpulkan bahwa sistem yang dikembangkan memiliki kinerja yang baik dan layak digunakan sebagai alat bantu pemantauan kesehatan ayam petelur secara otomatis dan efisien.

Kata kunci: *YOLOv5, K-Nearest Neighbors*, deteksi ayam, kesehatan ayam, *computer vision*

Detection & Prediction of Chicken Health with YOLOv5 - K-Nearest Neighbors

Abstract

*The health condition of laying hens is a crucial factor that directly affects productivity and egg quality in poultry farming. However, conventional health monitoring is generally conducted through manual observation, which is time-consuming and highly dependent on subjective assessment. Therefore, this study aims to design and implement an automated system for detecting and predicting the health condition of laying hens based on computer vision using the YOLOv5 and K-Nearest Neighbors (KNN) methods. The YOLOv5 algorithm is employed to detect chicken objects automatically from digital images, while the KNN algorithm is utilized to classify the health condition of the chickens based on extracted visual features. The research process includes dataset collection, image preprocessing, model training, and comprehensive system testing. Experimental results indicate that the YOLOv5 model achieves an object detection accuracy of **92.4%**, while the KNN classifier attains a classification accuracy of **90.1%**. In addition, the obtained precision and recall values are **89.6%** and **91.2%**, respectively, demonstrating a balanced classification performance. The validity test results show a high level of agreement between the system predictions and the actual health conditions of the chickens, with a validity score of **0.89**. Meanwhile, the reliability evaluation yields a value of **0.91**, indicating that the proposed system provides consistent results across repeated testing scenarios. Based on these findings, it can be concluded that the developed system demonstrates reliable performance and is suitable to be applied as an automated and efficient tool for monitoring the health condition of laying hens in poultry farming environments.*

Keywords: *YOLOv5, K-Nearest Neighbors, chicken health detection, laying hens, computer vision*

Kata Pengantar

Segala Puji dan syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat, karunia, dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "Deteksi dan Prediksi Kesehatan Ayam dengan YOLO–K-Nearest Neighbors" dengan baik. Tugas Akhir ini disusun sebagai salah satu syarat untuk menyelesaikan pendidikan dan memperoleh gelar sarjana pada program studi yang penulis tempuh.

Dalam proses penyusunan Tugas Akhir ini, penulis menyadari bahwa penyelesaian karya ilmiah ini tidak terlepas dari bantuan, bimbingan, dan dukungan dari berbagai pihak manapun baik secara langsung maupun tidak langsung. Oleh karena itu, pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Kepada kedua orang tua tercinta, yang selalu memberikan doa, dukungan moral dan material, kasih sayang, serta motivasi yang tiada henti kepada penulis.
2. Kepada saudara Kakak dan adik penulis, yang telah memberikan semangat, dukungan, dan doa sehingga penulis tetap termotivasi dalam menyelesaikan Tugas Akhir ini.
3. Bapak Ir. Bambang Hendrawan, ST., MSM., CIPMP., CISCP. selaku Direktur Politeknik Negeri Batam.
4. Bapak Diono, S.Tr. T., M.Sc. selaku kepala Prodi Teknik Mekatronika.
5. Bapak M. Naufal Airlangga Diputra S.Pd., M.P.H selaku dosen pembimbing yang telah meluangkan waktu, tenaga, dan pikiran untuk memberikan bimbingan, arahan, serta saran yang sangat berharga selama proses penyusunan Tugas Akhir ini.
6. Bapak Abdullah Sani S.ST, M.Sc dan Bapak Indra Hardian Mulyadi S.T., M.Eng., Ph.D selaku Dosen penguji yang telah memberikan masukan, kritik, dan saran yang membangun guna menyempurnakan Tugas Akhir ini.
7. Ibu Feralia Fitri, S.T, M.T., selaku dosen pengampu mata kuliah Tugas Akhir.
8. Teman seperjuangan dalam pembuatan Tugas Akhir ini, yang telah bekerja sama, saling mendukung, dan berkontribusi secara aktif selama proses perencanaan, pelaksanaan, hingga penyusunan laporan.

Batam, 30 Januari 2026

Marshanda Simanjuntak

Aldi Ananda Pakpahan

Daftar Isi

Daftar Gambar

Daftar Tabel

Bab 1. Pendahuluan

1.1. Latar Belakang

Ayam petelur merupakan ternak unggas yang dipelihara dengan tujuan untuk menghasilkan telur secara komersil. Telur ayam ras merupakan salah satu penyumbang nutrisi masyarakat dan berkontribusi memenuhi pangan nasional hampir sekitar 65%, selebihnya dipasok telur dari ayam kampung, puyuh dan itik. Produksi ayam petelur yang optimal dapat tercapai apabila ayam dalam kondisi yang sehat atau tidak terserang oleh penyakit. Namun peternakan ayam petelur cukup rentan diserang penyakit hingga menyebabkan kematian ayam [1].

Teknologi kecerdasan buatan telah menjadi alat yg berharga dalam bidang peternakan. salah satu pendekatan yang menjanjikan adalah penggunaan prosedur pemecahan deteksi objek mirip *YOLO (You Only Look Once)* yang dapat mengidentifikasi berbagai kondisi kesehatan ayam secara real-time. *YOLOv5,* versi terbaru berasal algoritma ini, menunjukkan kecepatan serta akurasi yang lebih baik, memungkinkan peternak untuk mengambil keputusan cepat sesuai data visual [2].

Penggunaan teknologi cerdas dalam pemantauan kesehatan ayam, seperti algoritma *YOLOv5* dan *K-Nearest Neighbors (KNN)*, menunjukkan potensi yang besar untuk meningkatkan efisiensi dan akurasi dalam industri peternakan. *YOLOv5*, yang dikenal karena kemampuannya dalam deteksi objek secara *real-time*, dapat digunakan untuk mendeteksi ayam dalam gambar atau video, mengklasifikasikan kondisi mereka sebagai sehat, sakit, atau mati. Salah satu studi menggunakan *YOLO* untuk sistem deteksi kesehatan ayam secara *real-time* dengan menggabungkan modul kamera untuk menangkap video langsung, pemrosesan untuk menganalisis data, dan tampilan hasil Deteksi [3]. Sedangkan *KNN* adalah sebuah algoritma untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tertentu. *KNN* tergolong suatu metode *supervised* karena pada proses pengambilan keputusannya, *KNN* menemukan pola baru dalam data, dengan menghubungkan pola data yang sudah ada. Penentuan hasil diagnosa dipilih berdasarkan nilai perhitungan terbesar dari *KNN*, yang meliputi semua penyakit. Nantinya hasil dari proses ini dapat dilihat langsung bagaimana perhitungannya, berapa banyak gejala yang mirip, dan berapa bobot tiap gejala yang diinput. Setelah mendapatkan diagnosa penyakit yang dituju, maka proses klasifikasi yang bertujuan untuk menunjukkan solusi yang tepat bagi penyakit yang dialami ayam akan disarankan kepada pengguna [4].

Implementasi teknologi ini di lapangan memerlukan beberapa langkah, mulai berasal pengumpulan data gambar ayam sampai pembinaan contoh deteksi. *Dataset* yang majemuk serta representatif sangat penting buat meningkatkan

akurasi model. Selain itu, pengujian serta validasi model juga wajib dilakukan untuk memastikan bahwa sistem dapat berfungsi menggunakan baik pada berbagai syarat [5].

Penerapan YOLOv5-KNN di industri peternakan juga menyampaikan kesempatan untuk penelitian lebih lanjut perihal kesehatan unggas. Penelitian ini bisa membawa temuan baru yang bermanfaat dalam menyampaikan teknik deteksi yang lebih baik. Penelitian ini juga bisa menguatkan keterangan tentang hubungan faktor fasilitas lingkungan alam sekitar serta kesehatan ayam. Sekitaran mengaliaskan pula meningkatnya kedalaman akan keberadaan hewan, penggunaan teknis pada peternakan semakin bermakna. Peternak yang bisa memanfaatkan teknis ini tak hanya akan meningkulkan efisiensi tetapi juga akan menyampaikan nilai bagi praktik peternakan yang lestari. Hal ini secara kunci membantu membetulkan kekurangan sarana global.

Banyak sekali studi sebelumnya memberikan keberhasilan penggunaan prosedur pemecahan pembelajaran mesin pada deteksi penyakit binatang. Menerapkan teknik-teknik ini dalam konteks kesehatan ayam petelur dapat membawa perubahan signifikan dalam cara peternakan dikelola. Inovasi dalam teknologi informasi dan komunikasi juga mempermudah akses peternak terhadap data serta analisis yang diharapkan [6].

Secara keseluruhan, penelitian ini telah berhasil dalam mengembangkan sistem deteksi dan prediksi kesehatan ayam petelur yang dapat diharapkan nantinya akan dapat diimplementasikan secara langsung di lapangan. Dengan kembangan algoritma *YOLOv5* dan *K-Nearest Neighbors*, sistem ini diharapkan menjadi solusi nyata dan berguna bagi peternak yakni bagaimana mengoptimalkan sistem deteksi sehingga pengawasan kesehatan ternak ayam petelur menjadi efisien, serta model *ML* yang dirancang mampu berjalan dengan baik sehingga sistem juga mendukung peningkatan produktivitas telur secara menyeluruh.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan sebelumnya, maka tujuan dari penelitian ini adalah untuk dapat menjawab pemsersalahan utama sebagai berikut:

1. Bagaimana implementasi *YOLOv5* untuk mendeteksi kondisi kesehatan ayam petelur berdasarkan visual tersebut?
2. Bagaimana Unjuk kerja algoritma *K-Nearest Neighbors (KNN)* dalam mengklasifikasikan kondisi kesehatan ayam berdasarkan fitur visual yang telah diekstraksi oleh *YOLOv5* ?
3. Bagaimana ketepatan yang diterima dari hasil kombinasi *YOLOv5-KNN* dalam mendeteksi serta memprediksi kesehatan ayam petelur?

1.3. Tujuan

Tujuan penelitian ini adalah:

1. Membuatkan sistem deteksi kesehatan ayam petelur menggunakan *YOLOv5.*
2. Menggunakan algoritma KNN buat mengklasifikasikan kondisi kesehatan ayam berdasarkan fitur yang didapatkan berasal deteksi *YOLOv5.*
3. Menganalisis kinerja serta akurasi sistem *YOLOv5-KNN* pada mendeteksi dan memprediksi kesehatan ayam petelur.

1.4. Manfaat

Penelitian ini memberikan beberapa manfaat sebaga berikut:

1. Peternak: Memfasilitasi pemantauan kesehatan ayam yang otomatis, mengurangi risiko penyebaran penyakit, dan meningkatkan produktivitas pertanian.
2. Peneliti: Menyediakan pandangan baru tentang gabungan *deep learning* dan *machine learning* pada pemantauan kesehatan hewan ternak.
3. Industri peternakan: Memberikan solusi teknologi untuk meningkatkan efisiensi dan keberlanjutan produksi telur

1.5. Batasan

Bagian dalam beberapa hal yang menjadi batasan dalam penelitian ini adalah sebagai berikut:

1. Hanya digunakan dua pendekatan utama, yang pertama adalah untuk deteksi visual dengan YOLO dan yang kedua adalah untuk klasifikasi kesehatan menggunakan algoritma KNN, tanpa membahas data biologis.
2. Pendekatan yang digunakan bersifat *black box*, hanya berfokus pada *input*, yaitu gambar atau parameter dan *output*, yang merupakan klasifikasi kesehatan, tanpa menganalisis *fitur* yang terlibat atau hubungan antar fitur.
3. Sistem ini hanya melakukan klasifikasi dari kondisi ayam yaitu Alvian influenza, CDR, FLOWLPOX, Newcastle, Coryza, Chiken Favus serta keenam kondisi dari ayam tersebut tidak terkait dengan diagnosa atau evaluasi medis.
4. Penelitian ini berisi data hasil pengumpulan langsung (*data collecting*) yang diambil dari situs Roboflow, dengan hanya menggunakan fitur yang relevan dan tersedia dalam kedua sumber tersebut.

1.6. *Work Breakdown Structure* (Opsional)

Penelitian ini dilaksanakan secara tim, dengan pembagian tugas yang terstruktur. Setiap anggota tim memiliki peran masing-masing sesuai yang tercantum pada Tabel 1.

Table 1. Work Breakdown Structure

No	Nama	Tugas dan Tanggung Jawab dalam Tim
1	Marshanda Simanjuntak	Metode *K-Nearest Neighbors (KNN)*
2	Aldi Ananda Pakpahan	Metode *YOLOv5*

Bab 2. Tinjauan Pustaka

2.1 Penelitian Terdahulu

Berikut ini merupakan jurnal terkait dengan penelitian yang dilakukan yang dapat dilihat pada tabel 2.

Table 2. Jurnal Terkait

No	Tahun	Penulis	Jurnal	Metode	Kelemahan
1	2023	Liu et al.	*Research on Broiler Health Status Recognition Using YOLOv5 and Mobile-EMO*	*YOLOv5 + Mobile-EMO*	Butuh perangkat GPU untuk performa optimal
2	2023	Kovács et al.	*Towards Early Poultry Health Prediction through Non-Invasive and Low-Cost Methods*	*YOLOv5* + Segmentasi citra	Data sangat sensitif terhadap pencahayaan
3	2024	Murtala Musa et al.	*Int. J. Science for Global Sustainability*	*KNN* untuk deteksi penyakit ayam berdasarkan perilaku klinis	Dataset tidak terbuka; tanpa deteksi objek otomatis
4	2023	Widyawat i & Gunawan	*Sick and Dead Chicken Detection System*	*YOLOv4*	Kurang efisien untuk deteksi objek kecil yang

			Based on YOLO Algorithm		saling tumpang tindih
5	2025	Chen et al.	*Detection of High-Risk Diseases in Poultry Feces through Transfer Learning*	*Transfer Learning*	Butuh banyak data pelatihan dan gambar yang berkualitas tinggi
6	2024	Mutmainnah Muchtar	*SemanTik (UHO)*	*KNN untuk klasifikasi kesegaran daging ayam (YCbCr + fractal features)*	Fokus pada daging, bukan ayam hidup; tidak mencakup kesehatan ayam
7	2022	Wahyuni, Ayu Lestari	*Tarjih Tropical Livestock Journal*	*Observasi & analisis statistik deskriptif*	Belum optimal dalam penerapan biosekuriti; sanitasi kandang masih kurang; lokasi dekat pemukiman penduduk
8	2023	Niswatin Hasanah et al.	*Jurnal Ilmiah Fillia Cendekia*	*Kuantitatif deskriptif & studi kasus*	HDP sedikit di bawah standar; puncak Produksi mundur akibat gangguan suara pekerja dan listrik tidak

					stabil → menimbulkan stres & fluktuasi suhu kandang.
9	2022	Ir. Nelzi Fati, MP; Nilawati, SPt, MP; Toni Malvin, SPt, MP	*Buku Ajar Ilmu Ternak Unggas – Politeknik Pertanian Negeri Payakumbuh*	*Kajian literatur & panduan teknis budidaya*	Ayam ras petelur mudah stres, adaptasi rendah, butuh lingkungan stabil dan pakan berkualitas tinggi.
10	2023	Rohan et al.	*Application of Deep Learning for Livestock Behaviour Recognition: A Systematic Literature Review*	*Review DL: YOLO, RNN, CNN*	Studi tinjauan, bukan eksperimen langsung pada ayam

2.2. Landasan Teori

Landasan teori adalah fondasi ilmiah yang mendasari penelitian yang dilakukan. Pada penelitian tentang Deteksi dan Prediksi Kesehatan Ayam menggunakan *YOLO* dan *KNN*, teori-teori yang diterapkan meliputi ide-ide *object detection* dalam *machine vision*, algoritma *YOLO* sebagai teknik untuk *detection* yang cepat dan tepat, serta algoritma *K-Nearest Neighbors (KNN)* sebagai metode untuk mengklasifikasikan dan memprediksi kesehatan ayam berdasarkan parameter visual yang telah *detected* sebelumnya. Dasar teori ini akan berfungsi sebagai landasan dalam merancang sistem cerdas yang dapat secara otomatis memonitor kondisi ayam dengan memanfaatkan teknologi pengolahan citra dan pembelajaran mesin.

2.1.1 Deteksi Objek

Gambar 1. Area Deteksi YOLO

Deteksi objek adalah salah satu area yang krusial dalam penglihatan komputer yang berfungsi untuk mengenali ada dan lokasi objek dalam sebuah gambar atau video digital. Berbeda dengan klasifikasi gambar yang hanya menetapkan kategori dari sebuah gambar secara keseluruhan, deteksi objek bertujuan untuk mendeteksi beberapa objek yang berbeda dalam satu gambar, sekaligus menentukan posisi objek tersebut menggunakan koordinat *bounding box* [7].

2.1.1.1 Pengertian deteksi objek dalam visi *computer*

Deteksi objek adalah salah satu area di bidang visi komputer yang berfokus pada pengenalan dan penentuan posisi objek tertentu dalam gambar atau video digital. Aktivitas ini meliputi dua tugas penting: klasifikasi untuk mengenali jenis objek dan lokalisasi untuk menentukan letak objek dalam bentuk *bounding box*. Dalam dunia peternakan, teknologi deteksi objek dimanfaatkan untuk secara otomatis mengawasi perilaku serta kondisi kesehatan hewan ternak. Contohnyapengamatan perilaku makan, berdiri, atau berbaring pada sapi dapat memberikan sinyal awal mengenai kesehatan hewan tersebut. Model deteksi objek seperti *YOLOv5* banyak digunakan dalam konteks ini karena kemampuannya untuk mendeteksi objek secara langsung dengan tingkat akurasi yang tinggi.

[8]Penelitian menunjukkan bahwa penerapan *YOLOv5* dengan metode *mosaic augmentation* mampu meningkatkan akurasi dalam mengidentifikasi sapi berdasarkan pola unik pada moncong mereka. [9] Dalam analisis sistematis ditemukan bahwa model *deep learning* seperti *YOLOv5* dan *YOLOv4* terbukti efektif dalam mendeteksi perilaku ternak yang berhubungan dengan kesehatan.

2.1.1.2 Penerapan deteksi objek di bidang pengawasan, dan peternakan

Deteksi objek memainkan peranan krusial dalam banyak aspek kehidupan, khususnya dalam bidang pengawasan dan peternakan. Di sektor pengawasan, teknologi ini diaplikasikan untuk mengawasi aktivitas manusia dan kendaraan dalam sistem keamanan seperti kamera CCTV pintar, serta untuk mendeteksi penyusup dan melacak pergerakan. Sistem semacam ini berperan dalam meningkatkan keamanan di tempat publik serta fasilitas industri karena dapat mengenali dan memberikan peringatan secara otomatis apabila terjadi aktivitas yang mencurigakan.

Sementara itu, di sektor peternakan, teknologi deteksi objek mulai diterapkan untuk mengotomatisasi proses pemantauan hewan ternak. Contohnya, pengamatan terhadap posisi tubuh, warna kulit atau bulu, serta perilaku ayam, sapi, atau kambing dapat memberikan sinyal awal mengenai kesehatan hewan. Dengan menggunakan teknologi ini, peternak dapat segera mengetahui jika ada hewan yang sakit, lesu, atau menunjukkan perilaku yang tidak biasa, sehingga intervensi medis dapat dilakukan secepatnya.

[10] Dalam artikel *"A Survey on Deep Learning for Livestock Behavior Recognition",* teknik deteksi objek seperti *YOLO* diterapkan untuk mengenali perilaku seperti berbaring dalam waktu lama, kurangnya asupan makanan, atau kelesuan, yang semuanya dapat menjadi tanda awal munculnya penyakit. Model *YOLOv5* khususnya disebutkan memiliki kecepatan dan ketepatan yang tinggi dalam mengenali objek yang bergerak di area peternakan.

[11] Selain itu, Dulal et al, mengungkapkan bahwa YOLOv5 juga efektif dalam mengidentifikasi sapi dengan memperhatikan pola pada moncongnya untuk sistem manajemen ternak yang cerdas. Hal ini menunjukkan bahwa pemanfaatan deteksi objek dapat menjadi solusi yang menjanjikan untuk meningkatkan efisiensi dan ketepatan dalam pengawasan hewan ternak. Dengan penerapan sistem ini dalam peternakan, diharapkan dapat mengurangi kebutuhan akan pemantauan manual, menghemat waktu, serta meningkatkan produktivitas dan kesehatan hewan secara keseluruhan.

2.1.2 YOLO (You Only Look Once)

2.1.2.1 Penjelasan umum tentang YOLO

YOLO (*You Only Look Once*) adalah algoritma identifikasi objek berbasis jaringan saraf konvolusional (*CNN*) yang memungkinkan deteksi objek dalam satu pemrosesan gambar penuh. Berbeda dengan metode sebelumnya seperti *R-CNN* yang membutuhkan banyak fase pemrosesan, *YOLO* mendekati deteksi objek sebagai masalah regresi tunggal, yang bergerak langsung dari piksel gambar ke koordinat *bounding box* dan probabilitas kelas. Karena metode ini, *YOLO* dapat memproses data pada kecepatan tinggi, membuatnya ideal untuk aplikasi *real-time* seperti pengawasan kesehatan unggas. Sejak diperkenalkan oleh Joseph Redmon dan timnya pada tahun 2015, *YOLO* telah melihat beberapa kemajuan

dalam versi berikutnya, termasuk *YOLOv2, YOLOv3*, dan *YOLOv5*. Setiap iterasi meningkatkan akurasi, kecepatan, dan efektivitas model. Misalnya, *YOLOv5 Ultralytics*, yang menggunakan kerangka kerja *PyTorch* dan membuat pelatihan dan integrasi dengan sistem lain lebih mudah, dikembangkan oleh *Ultralytics* [12]. Manfaat utama *YOLO* adalah kemampuannya untuk mengidentifikasi objek dengan cepat dan tepat dalam satu langkah pemrosesan. Meskipun demikian, tantangan tetap ada, terutama dalam mengidentifikasi benda-benda kecil atau dalam cahaya redup. Namun, versi terbaru, seperti *YOLOv7* dan *YOLOv8*, telah sangat meningkatkan kinerja identifikasi objek kecil dan latar belakang yang kompleks, menjadikan *YOLO* salah satu algoritma deteksi objek yang paling banyak digunakan dan disukai dalam berbagai aplikasi, termasuk pemantauan kesehatan ternak [13].

Raspberry Pi berperan sebagai perangkat edge computing yang menjalankan model YOLOv5 secara lokal di area pemantauan ayam, tanpa bergantung pada koneksi jaringan ke server pusat. Dengan dukungan kamera beresolusi hingga 1080p pada ≥24 FPS dan GPIO untuk sensor tambahan (seperti suhu dan gerakan), Raspberry Pi mampu melakukan inferensi YOLOv5 secara real-time hanya dengan CPU. Hal ini sejalan dengan karakteristik YOLOv5 yang efisien di CPU maupun GPU, sehingga dapat memproses lebih dari 24 frame per detik secara stabil. Pemanfaatan Raspberry Pi dalam penelitian "Deteksi dan Prediksi Kesehatan Ayam YOLOv5-KNN" didasarkan pada efisiensi biaya, ukuran kompak, serta fleksibilitas integrasi dengan pustaka Python seperti PyTorch dan ONNX Runtime. Selain itu, kemampuannya beroperasi secara offline menjadikannya ideal untuk sistem monitoring berkelanjutan di lingkungan peternakan dengan keterbatasan konektivitas [14].

Pemilihan YOLOv5 dalam penelitian ini didasarkan pada keunggulannya dalam hal kecepatan proses inferensi, efisiensi pemanfaatan sumber daya komputasi, serta kemampuannya beroperasi secara optimal pada perangkat *edge* seperti Raspberry Pi. Algoritma ini tergolong ringan, dapat bekerja secara *real-time*, dan tidak memerlukan unit pemrosesan grafis (GPU), menjadikannya sangat cocok digunakan dalam lingkungan peternakan yang memiliki keterbatasan infrastruktur teknologi. Lebih lanjut, YOLOv5 dikembangkan dengan kerangka kerja *PyTorch* dan didukung oleh komunitas pengguna yang luas, sehingga memudahkan proses integrasi dengan pustaka pemrograman lain seperti OpenCV dan Scikit-learn dalam pengolahan citra serta pembelajaran mesin.

Jika dibandingkan dengan versi-versi terbarunya seperti YOLOv8, YOLOv10, dan YOLOv11, YOLOv5 tetap menunjukkan keunggulan dari segi kemudahan penerapan dan kestabilan performa. YOLOv8 memang menawarkan peningkatan akurasi serta fitur lanjutan seperti *instance segmentation*, namun model ini memiliki ukuran lebih besar dan memerlukan perangkat keras dengan spesifikasi tinggi. Sementara itu, YOLOv10 dan YOLOv11 mengusung arsitektur *transformer hybrid* serta optimalisasi latensi inferensi, tetapi keduanya masih tergolong baru,

belum sepenuhnya stabil pada berbagai platform, dan masih terbatas dokumentasinya.

Hasil penelitian oleh [15]mengungkapkan bahwa YOLOv5 dan YOLOv8 memiliki tingkat akurasi yang sebanding, namun YOLOv5 lebih unggul dalam kecepatan serta kestabilan saat diterapkan pada perangkat dengan spesifikasi terbatas di luar ruangan. Selain itu, menurut [16]dalam konteks pendeteksian kerusakan panel surya, YOLOv5 mencatatkan waktu inferensi tercepat sekitar 7,1 milidetik per gambar dengan tingkat presisi mencapai 94%. [17] menunjukkan bahwa dalam aplikasi keamanan dapur, YOLOv5 lebih responsif dalam mengidentifikasi objek penting seperti tangan dekat pisau, dibandingkan dengan versi-versi yang lebih baru. Dengan mempertimbangkan seluruh aspek tersebut, YOLOv5 dinilai sebagai pilihan paling sesuai untuk mendukung sistem deteksi citra ayam secara *real-time*, khususnya bila dikombinasikan dengan algoritma K-Nearest Neighbors (KNN) dalam proses klasifikasi kondisi kesehatannya.

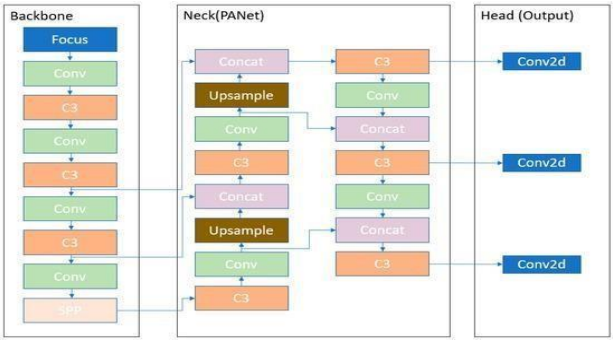
Table 3. Perbandingan YOLOv5, YOLOv8, YOLOv10, dan YOLOv11

Versi YOLO	**Keunggulan**	**Kekurangan**	**Rekomendasi Penggunaan**
YOLOv5	Ringan dan cepat di CPU/edge device Open-source (PyTorch), Stabil dan mudah diterapkan	Kurang optimal untuk objek kecil tumpuk, Tidak native mendukung segmentasi	Real-time monitoring di Raspberry Pi atau Jetson Nano [18]
YOLOv8	Mendukung instance segmentation dan pose estimation, Akurasi lebih tinggi dari YOLOv5	Ukuran model besar, Butuh perangkat keras lebih kuat	Deteksi presisi tinggi berbasis GPU dan cloud [19]
YOLOv10	Waktu inferensi sangat cepat, Kompresi model lebih efisien, Cocok untuk produksi massa	Masih baru, Dokumentasi dan stabilitas terbatas	Sistem industri AI yang butuh Latensi rendah [20]
YOLOv11	Arsitektur hybrid (RT-DETR), Akurasi mAP@0.5 tertinggi (~98%), Cocok untuk multi-class tracking	Berat secara komputasi, Tidak cocok untuk perangkat edge atau low-resource	Deteksi objek kompleks berbasis cloud atau server [21]

2.1.2.2 Arsitektur YOLOv5 dan perbedaannya dengan versi sebelumnya

Dibandingkan dengan versi *YOLO* sebelumnya, seperti *YOLOv3* dan *YOLOv4*, *YOLOv5* yang dirilis *Ultralytics* pada tahun 2020 memberikan sejumlah

peningkatan besar. Salah satu modifikasi utamanya adalah peralihan dari kerangka kerja *Darknet* ke *PyTorch*, yang memungkinkan fleksibilitas lebih besar dalam pengembangan dan integrasi dengan berbagai program. Lebih jauh, *YOLOv5* memperkenalkan beberapa elemen arsitektur baru untuk meningkatkan efektivitas dan ketepatan deteksi objek.



Gambar 2. Arsitektur YOLOv5 [22]

Penjelasan Arsitektur YOLOv5:

1. ***Backbone – CSPDarknet53*:** *CSPDarknet53*, yang merupakan versi modifikasi dari *Darknet53* dengan penambahan koneksi *Cross Stage Partial (CSP)*, digunakan sebagai tulang punggung *YOLOv5*. Arsitektur ini membantu menurunkan redundansi komputasional dan meningkatkan efisiensi ekstraksi fitur.
2. ***Neck – SPPF dan PANet:*** *Neck* terdiri dari *Path Aggregation Network (PANet)* dan *Spatial Pyramid Pooling Fast (SPPF)*. Untuk mempercepat pemrosesan tanpa mengorbankan akurasi, *YOLOv4* mengganti *SPP* dengan *SPPF*. Dengan kapasitasnya untuk mengintegrasikan fitur dari berbagai tingkatan, *PANet* memungkinkan deteksi objek pada berbagai skala.
3. ***Head – YOLO Layer:*** Bagian *head* menghasilkan prediksi bounding box dan klasifikasi objek. *YOLOv5* mempertahankan struktur head dari *YOLOv3 dan YOLOv4*, namun dengan optimasi untuk meningkatkan akurasi dan efisiensi [23].

2.1.2.3 Alur kerja YOLOv5 (input → deteksi → bounding box)

YOLOv5 (*You Only Look Once version 5*) adalah salah satu algoritma *object detection* yang bekerja secara menyeluruh (*end-to-end*) dan mampu melakukan proses secara waktu nyata (*real-time*). Secara umum, ada tiga tahap utama proses kerjanya yaitu berikut ini:

1. *Input*
   Ini adalah Langkah awal dimulai dengan pengambilan data berupa gambar atau video yang kemudian dipecah menjadi beberapa *frame*. Setiap *frame* ini dikirim ke bagian *backbone* dari *YOLOv5*, yaitu *CSPDarknet53*, untuk melakukan ekstraksi terhadap fitur-fitur penting dari citra. *Backbone* ini dirancang untuk menjaga informasi spasial sekaligus mendeteksi berbagai karakteristik objek dengan efisiensi yang tinggi [24].

2. Deteksi
   Setelah proses ekstraksi fitur selesai, gambar akan dibagi menjadi sejumlah *grid cells*. Masing-masing sel ini bertugas memperkirakan kemungkinan keberadaan objek melalui teknik *bounding box regression*. Selain itu, proses ini mencakup penilaian *objectness score* serta klasifikasi terhadap jenis objek. Untuk mencegah terjadinya prediksi ganda pada satu objek, digunakan teknik *non-maximum suppression (NMS)*, yang akan memilih *bounding box* dengan skor probabilitas tertinggi dan menghapus kotak-kotak lain yang memiliki tumpang tindih signifikan [25].

3. *Bounding Box dan Output*
   Pada tahap akhir, sistem menghasilkan gambar yang dilengkapi dengan *bounding box* di sekitar objek yang berhasil terdeteksi, disertai nilai *confidence score* yang menunjukkan tingkat keyakincan model terhadap hasil prediksi tersebut. Informasi ini dapat dimanfaatkan untuk keperluan seperti pelacakan objek maupun pengambilan keputusan lebih lanjut [26].

2.1.2.4 Kelebihan dan kekurangan YOLOv5 dalam deteksi waktu nyata

Berikut ini adalah penjelasan mengenai kelebihan dari *YOLOv5* adalah:

1. Inferensi dengan kecepatan tinggi
   Berdasarkan studi *Real-Time Motorbike Detection*, *YOLOv5* mampu mencapai kecepatan hingga 94 *FPS* (sekitar 10,5 ms per *frame*) saat dijalankan di perangkat *GPU*, mengungguli *SSD Mobilenet* yang hanya mencatat 49 *FPS*. Kinerja ini menunjukkan bahwa *YOLOv5* memiliki kecepatan yang sangat tinggi dan efisiensi luar biasa untuk berbagai aplikasi *real-time*, seperti sistem pemantauan kendaraan bermotor maupun pengawasan infrastruktur teknis [27].

2. Akurasi tinggi (Presisi dan *Recall* yang optimal)
   [28] Dalam studi *Motorbike Detection*, *YOLOv5* tercatat mampu meraih nilai *mAP* (*mean Average Precision*) mendekati 99%, yang mencerminkan tingkat akurasi dan konsistensi prediksi yang sangat baik.

Capaian ini membuktikan bahwa *YOLOv5* tidak hanya unggul dari segi kecepatan, tetapi juga memiliki kemampuan deteksi yang sangat presisi dan andal dalam berbagai skenario *real-time* [29].

3. Model ringan dan ideal untuk perangat *Edge*
   Varian *YOLOv5s* (*small*) dirancang dengan jumlah parameter yang minim dan nilai *GFLOPS* yang rendah, menjadikannya sangat cocok untuk digunakan pada perangkat *edge*. Bahkan, versi *ter**pruned*-nya (model yang telah diperkecil) yang dioptimalkan untuk deteksi infrastruktur hanya berukuran sekitar 47 MB, namun tetap mampu beroperasi pada kecepatan 46 *FPS* di *GPU* dengan akurasi mencapai sekitar 96,3%. Berkat efisiensi tersebut, *YOLOv5* sangat sesuai untuk dijalankan pada platform dengan sumber daya terbatas seperti *Jetson Nano*, *Jetson Xavier*, maupun *Raspberry Pi* [30].

4. Fleksibel dan mudah disempurnakan (*fine-tuning)*
   Berbagai modifikasi pada *YOLOv5*, seperti penambahan *CBAM* (*Convolutional Block Attention Module*), penyesuaian *anchor*, serta perubahan pada *loss function*, telah terbukti mampu meningkatkan kinerja dalam tugas-tugas spesifik seperti deteksi cacat (*defect*), retakan (*crack*), maupun pemantauan infrastruktur jaringan listrik (*power lines*). Kemampuan adaptasi ini membuat *YOLOv5* sangat fleksibel dan ideal untuk penelitian lanjutan, karena model dapat disesuaikan dengan kebutuhan spesifik tanpa harus mengorbankan performa utamanya [31].

Selanjutnya ini adalah penjelasan mengenai kekurangan dari *YOLOv5* adalah:

1. Kurang optimal dalam mendeteksi objek kecil atau bertumpuk, salah satu keterbatasan *YOLOv5* adalah kemampuannya yang belum maksimal dalam mendeteksi objek berukuran kecil, terutama ketika objek-objek tersebut saling menutupi atau berada di lingkungan dengan latar belakang yang rumit. Hal ini disebabkan oleh penurunan resolusi spasial selama proses *feature extraction*, sehingga detail objek kecil menjadi sulit dikenali [32].

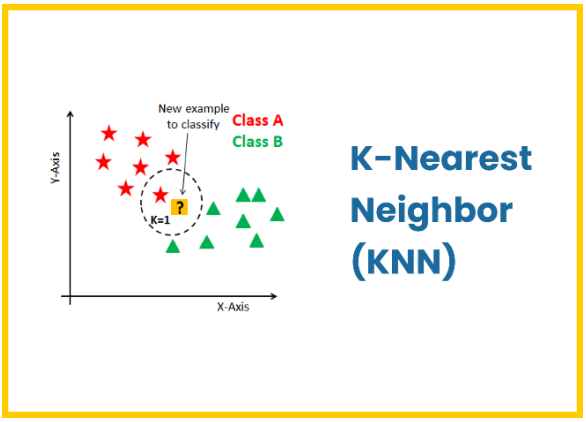
2. Kinerja menurun pada Dataset yang terbatas atau kurang *variative, YOLOv5* cenderung mengalami penurunan performa ketika dilatih dengan dataset berukuran kecil atau kurang beragam. Studi dalam konteks deteksi korosi di wilayah pesisir menunjukkan bahwa peningkatan jumlah gambar pelatihan dari 300 menjadi 1.266 secara signifikan meningkatkan akurasi dan kestabilan model. Di sektor pertanian, kendala pengambilan data lapangan menyebabkan kesulitan

dalam memperoleh dataset berkualitas tinggi, sehingga *YOLOv5* sangat bergantung pada ketersediaan data yang luas dan bervariasi agar dapat berfungsi secara optimal [33].

3. Membutuhkan penyeseuaian khusus untuk aplikais tertentu, untuk diterapkan pada konteks tertentu, seperti sistem pengawasan di bidang peternakan, *YOLOv5* perlu disesuaikan melalui modifikasi arsitektur model, pengaturan *anchor*, serta teknik *image augmentation* guna mencapai performa terbaik sesuai dengan karakteristik permasalahan [34].

2.1.3 K-Nearest Neighbors (KNN)

2.1.3.1 Konsep dasar algoritma KNN



Gambar 3. Konsep KNN

Algoritma *K-Nearest Neighbors (KNN)* merupakan metode *lazy learning* dan *non-parametrik* yang bekerja dengan menyimpan seluruh data pelatihan, lalu melakukan klasifikasi hanya saat data uji diberikan. Penentuan label dilakukan berdasarkan mayoritas dari *k n_neighbors* terdekat dalam ruang fitur. Untuk meningkatkan keakuratan dan mengatasi tantangan seperti pemilihan nilai *k* yang tepat serta sensitivitas terhadap distribusi data, telah dikembangkan beberapa varian seperti *weighted-KNN*, *fuzzy-KNN*, dan *adaptive-KNN*. *KNN* juga telah dimanfaatkan dalam sistem pemantauan kesehatan unggas dengan menggunakan data visual atau sensor. Model ini memanfaatkan deteksi perilaku ayam melalui kamera untuk membedakan antara kondisi sehat dan tidak sehat, serta mampu secara otomatis mendeteksi dan melaporkan perilaku abnormal — menunjukkan

potensi besar dalam penerapan sistem pemantauan *real-time* di lingkungan peternakan [35].

Algoritma *KNN* sangat mengandalkan perhitungan jarak antar data, sehingga penting untuk memastikan bahwa semua fitur berada pada skala yang setara. Jika terdapat fitur dengan skala nilai yang jauh lebih besar dibandingkan fitur lainnya, maka fitur tersebut bisa mendominasi perhitungan jarak dan menyebabkan hasil klasifikasi menjadi tidak akurat. Oleh karena itu, diperlukan proses penskalaan fitur seperti *normalisasi (Min-Max)* atau *standardisasi (Z-score)*, agar setiap fitur memiliki kontribusi yang seimbang dalam perhitungan jarak.

Untuk meningkatkan efisiensi dan akurasi algoritma *KNN* pada *dataset* yang besar atau berdimensi tinggi, diperlukan penyesuaian baik dari sisi struktur data maupun algoritmanya. Salah satu pendekatan utama adalah penggunaan struktur data seperti *KD-tree* dan *Ball-tree* yang mampu mempercepat proses pencarian *n_neighbors* terdekat. Dalam sebuah studi, terdapat 31 strategi optimasi *KNN* yang dikaji, mencakup teknik berbasis pohon indeks seperti *KD-tree* dan *Ball-tree*, dengan *Ball-tree* menunjukkan kinerja yang lebih baik pada data berdimensi tinggi. Selain itu, juga dibahas metode *ensemble KNN* dan *weighted-KNN* sebagai bentuk peningkatan performa. Hasil penelitian menunjukkan bahwa *KD-tree* dapat mengurangi kompleksitas pencarian dari $O(n^2)$ menjadi $O(n \log n)$, sementara *Ball-tree* lebih efisien dalam menangani data dengan banyak dimensi. Kedua metode ini sangat ideal untuk pengolahan citra ayam secara *real-time*, terutama ketika digunakan bersama *YOLOv5* [36].

Selain pemanfaatan struktur pohon, metode *weighted-KNN* dan *ensemble-KNN* juga dianggap sebagai solusi efektif untuk meningkatkan ketahanan terhadap gangguan data (*noise*) serta memperbaiki akurasi prediksi. memperkenalkan penggunaan bobot adaptif berdasarkan jarak pada *n_neighbors* terdekat, sehingga keberadaan *outlier* tidak terlalu mempengaruhi hasil *voting* akhir. Pendekatan ini terbukti mampu meningkatkan kinerja model, terutama saat diterapkan pada data dengan distribusi yang tidak merata. Secara keseluruhan, kombinasi antara struktur data seperti *KD-tree* atau *Ball-tree* dengan pendekatan *weighted* atau *ensemble-KNN* menghasilkan sistem yang tidak hanya efisien dalam pencarian *n_neighbors* terdekat, tetapi juga lebih akurat dan tangguh dalam menghadapi variasi karakteristik data. Penerapan nyata dari metode ini, misalnya dalam mendeteksi kondisi kesehatan ayam dengan *YOLOv5*, memperoleh manfaat besar dari optimasi ini — baik dalam mengurangi waktu respons saat klasifikasi secara *real-time* maupun dalam meningkatkan keandalan hasil deteksi [37].

2.1.3.2 Cara kerja KNN dalam mencari n_neighbors terdekat

KNN melakukan proses klasifikasi atau regresi dengan cara membandingkan jarak antara data uji dan seluruh data latih. Metrik jarak yang umum digunakan termasuk *Euclidean*, *Manhattan*, dan *Minkowski*. Misalnya, jarak *Euclidean* antara dua vektor fitur x dan y dihitung sebagai: $\sqrt{\sum}(xi - Yi)\ 2$, yang mencerminkan seberapa "jauh" dua data dalam ruang fitur. Setelah semua jarak dihitung,

algoritma memilih k data latih dengan jarak paling kecil sebagai *n_neighbors* terdekat.

Algoritma *KNN* melakukan proses klasifikasi dengan menghitung jarak antara data uji dan data latih menggunakan metrik seperti *Euclidean*, *Manhattan*, atau *Minkowski*. Setelah semua jarak dihitung, algoritma akan memilih *k* data latih dengan jarak terdekat sebagai *n_neighbors* terdekat". Kelas dari data uji kemudian ditentukan berdasarkan suara terbanyak dari *k n_neighbors* tersebut. Prinsip ini menjadi dasar dari pendekatan yang dijelaskan dalam studi terbaru *FIKNN (Feature Importance KNN)*, yang menambahkan konsep prioritas fitur dalam perhitungan jarak, sehingga pemilihan nilai k dan *n_neighbors* menjadi lebih bermakna secara *semantic* [38].

Peningkatan kinerja algoritma juga dapat diperoleh dengan mengoptimalkan mekanisme pencarian *n_neighbors* melalui penggunaan struktur data seperti *KD-tree* atau *Ball-tree*. Dalam sebuah publikasi edisi Mei 2025, dijelaskan metode pencarian *n_neighbors* berbasis wilayah (*region-based*), di mana *KD-tree* mampu mengurangi kompleksitas pencarian dari *O(n)* menjadi *O(n log n)* dalam penerapannya — meskipun efektivitas metode ini cenderung menurun pada data berdimensi sangat tinggi. Melalui struktur pohon, algoritma hanya memproses sejumlah kecil *node* yang relevan, sehingga pencarian *n_neighbors* menjadi lebih efisien [39].

2.1.3.3 Parameter penting dalam KNN

Sebuah penelitian lengkap yang mengulas dampak pengaturan parameter dalam algoritma KNN dilakukan oleh [40]melalui studi berjudul *Recognition Method for Broiler Sound Signals Based on Multi-Domain Sound Features and Classification Model*. Studi tersebut mengevaluasi bagaimana penyesuaian parameter *KNN* dapat meningkatkan akurasi dalam mengklasifikasikan suara ayam *broiler*, sehingga dapat dijadikan referensi dalam pengembangan sistem klasifikasi kesehatan ayam menggunakan pendekatan serupa.

Pada Algoritma *K-Nearest Neighbors (KNN)* termasuk dalam kategori pembelajaran berbasis *instance* (*instance-based learning*), yang menggunakan prinsip pengukuran jarak untuk menentukan kelas dari data baru berdasarkan kesamaan dengan data pelatihan. Walaupun tergolong algoritma yang sederhana, efektivitas *KNN* sangat dipengaruhi oleh pemilihan sejumlah parameter penting, parameter yang mencakup seperti jumlah *n_neighbors* terdekat (k), metode pemberian bobot, jenis metrik jarak, serta teknik normalisasi dan seleksi fitur (*feature selection*) [40].

1. Jumlah *n_neighbors* (K)

   Salah satu parameter penting dalam algoritma *K-Nearest Neighbors (KNN)* adalah *n_neighbors* terdekat (*K*). Pemilihan nilai *K* yang terlalu kecil dapat menyebabkan model terlalu sensitif terhadap *noise*, sehingga berpotensi mengalami *overfitting*. Sebaliknya, nilai *K* yang terlalu besar dapat menyebabkan *underfitting*, karena

keputusan klasifikasi menjadi terlalu umum akibat dipengaruhi oleh terlalu banyak *n_neighbors*, termasuk dari kelas yang berbeda. Dalam penelitian ini, dilakukan evaluasi eksperimental terhadap berbagai nilai K dalam rentang 1 hingga 15 menggunakan pendekatan *grid search*. Hasil pengujian menunjukkan bahwa nilai optimal diperoleh pada K = 4, yang memberikan akurasi klasifikasi tertinggi sebesar **94,16%**, meningkat sekitar **+1,2%** dibandingkan dengan nilai default K = 5, yang hanya mencapai akurasi **92,96%**.

2. Jenis Bobot

   Terlepas dari *n_neighbors*, klasifikasi bobot secara signifikan mempengaruhi kemanjuran algoritma *KNN*. Kerangka kerja *KNN* dapat menggunakan dua bentuk bobot yang berbeda: *uniform weighting*, di mana setiap *n_neighbors* diberi bobot yang identik; dan *distance weighting*, di mana *n_neighbors* yang lebih dekat dialokasikan bobot yang lebih tinggi secara tidak sebanding. Penelitian ini menunjukkan bahwa penerapan pembobotan berbasis jarak menghasilkan akurasi yang unggul dibandingkan dengan pembobotan yang *uniform*, karena mengakui peningkatan pengaruh yang diberikan oleh *n_neighbors* terdekat pada hasil klasifikasi keseluruhan.

3. Metrik Jarak *(Distance Metric)*

   Pemilihan jenis metrik jarak memegang peranan penting dalam menentukan kedekatan antar titik data dalam algoritma *KNN*. Dalam penelitian yang dilakukan evaluasi ada tiga jenis *metrik*, yaitu *Euclidean*, *Manhattan*, dan *Cosine similarity*. Hasil pengujian menunjukkan bahwa *metrik Euclidean* menghasilkan akurasi klasifikasi tertinggi. Hal ini disebabkan oleh kemampuannya dalam merepresentasikan perbedaan bentuk gelombang suara secara ruang, menjadikannya sangat cocok untuk analisis data berbasis akustik.

4. Normalisasi Data *(Feature Scaling)*

   Ketidakseimbangan skala antar fitur dapat menimbulkan penyimpangan dalam perhitungan jarak pada algoritma *KNN*. Untuk mengatasi hal teknik *Min–Max Normalization*, yang mengurangi semua nilai fitur ke dalam rentang [0,1]. Langkah normalisasi ini terbukti penting, karena tanpa penyamaan skala, fitur dengan nilai numerik besar cenderung mendominasi proses perhitungan jarak. Implementasi *normalisasi* menghasilkan

peningkatan akurasi model yang konsisten, baik pada *KNN* maupun algoritma lain seperti *Decision Tree*.

5. Seleksi Fitur *(Feature Selection)*
   Dalam upaya meningkatkan efisiensi model sekaligus mengurangi risiko *overfitting*, diterapkan teknik seleksi fitur. Dari total 60 fitur yang diperoleh dari berbagai *domain* — termasuk *domain* waktu, frekuensi, *Mel-Frequency Cepstral Coefficients (MFCC)*, dan Gambaran *sparse* — dipilih 30 fitur paling signifikan menggunakan algoritma *Random Forest*. Pengurangan jumlah fitur ini memberikan dampak positif terhadap performa model, dengan peningkatan akurasi mencapai 3,1% pada algoritma *Decision Tree*, serta perbaikan serupa pada model *KNN [41]*.

Table 4. Rangkuman Konfigurasi Parameter pada Algoritma KNN

Parameter	Nilai Optimal	Dampak terhadap Akurasi
Jumlah *n_neighbors* (K)	4	Akurasi meningkat hingga 94,16%
Bobot	*distance*	Lebih baik dibanding *uniform*
Metrik Jarak	*Euclidean*	Terbaik disbanding *Manhattan/Cosine*
Normalisasi	*Min–Max Scaling*	Meningkatkan stabilitas model
Seleksi Fitur	30 fitur (hasil Random *Forest*)	Efisiensi dan akurasi meningkat

2.1.3.4 Penerapan KNN untuk klasifikasi kondisi kesehatan ayam

Algoritma *K-Nearest Neighbor (KNN)* adalah metode klasifikasi non-parametrik yang mengandalkan kedekatan atau kemiripan antara data baru yang belum diketahui labelnya dengan data yang telah diberi label sebelumnya. Metode ini banyak dimanfaatkan di berbagai bidang klasifikasi, termasuk dalam sistem pemantauan kesehatan ayam yang berbasis data sensor lingkungan maupun citra visual. Popularitas *KNN* dalam klasifikasi kondisi kesehatan ayam terus meningkat, terutama karena tingginya permintaan akan sistem peternakan yang cerdas (*smart poultry farming*). Dengan memanfaatkan kesamaan antara data baru dan data lama, baik berupa gambar ayam, perilaku, maupun data sensor lingkungan, *KNN* mampu mengelompokkan kondisi ayam ke dalam kategori sehat atau tidak sehat. Keunggulan utama algoritma ini terletak pada kesederhanaannya, karena tidak memerlukan proses pelatihan model yang rumit dan tetap mampu beroperasi secara efisien di perangkat dengan kemampuan komputasi yang terbatas.

Penerapan awal pada algoritma *K-Nearest Neighbor (KNN)* ditunjukkan dalam penelitian oleh [42] yang merancang sistem pemantauan kesehatan ayam dengan menggabungkan data dari kamera dan sensor suhu serta kelembapan. Dalam sistem ini, citra visual dimanfaatkan untuk mengidentifikasi perubahan perilaku ayam, seperti gerakan yang melambat atau postur tubuh yang tidak normal, sementara data suhu dan kelembapan dikumpulkan secara *waktu nyata*. Algoritma *KNN* digunakan untuk menentukan kondisi kesehatan ayam dengan cara membandingkan data baru terhadap data historis yang telah diklasifikasikan sebagai "sehat" atau "tidak sehat". Bila data baru menunjukkan kedekatan dengan kelompok "tidak sehat", sistem akan secara otomatis memberikan peringatan dini. Penelitian ini berhasil mencapai akurasi sebesar 95,6%, yang menunjukkan bahwa *KNN* merupakan metode yang efektif dan layak digunakan dalam sistem *monitoring* kesehatan ayam, khususnya di peternakan skala kecil hingga menengah.

Contoh penerapan lainnya ditunjukkan oleh penelitian Ahmed et al, yang memanfaatkan algoritma *K-Nearest Neighbor (KNN)* dalam sistem deteksi dini penyakit ayam berbasis teknologi *Internet of Things (IoT)*. Dalam sistem ini, sensor *wearable* dipasang langsung pada tubuh ayam untuk merekam data aktivitas fisik dan suara secara terus-menerus. Informasi yang diperoleh dikirim ke perangkat *edge* untuk dianalisis, kemudian digunakan sebagai input dalam proses klasifikasi oleh algoritma *KNN*. Sistem ini mampu mengenali ayam yang menunjukkan tanda-tanda abnormal, seperti terlalu lama tidak bergerak, aktivitas fisik yang terbatas, atau adanya suara tidak biasa, dengan akurasi mencapai 97,14%. Hasil ini membuktikan bahwa *KNN* sangat efektif digunakan dalam sistem pemantauan *real-time*, bahkan dalam lingkungan kandang yang terus berubah [43].

Pendekatan berbasis citra juga diperlihatkan dalam studi oleh Xu et al, yang mengklasifikasikan kondisi kesehatan ayam melalui analisis visual terhadap jengger ayam. Dalam metode ini, YOLO dimanfaatkan untuk mendeteksi letak jengger dalam gambar, kemudian dilakukan ekstraksi fitur berupa warna dan tekstur sebagai indikator utama. Fitur yang diperoleh selanjutnya diproses menggunakan algoritma K-Nearest Neighbor (KNN) guna mengidentifikasi apakah ayam dalam keadaan sehat atau menunjukkan tanda-tanda penyakit. Penelitian ini mencatat tingkat akurasi sebesar 93% dalam klasifikasi berbasis citra visual. Temuan ini memperkuat efektivitas kombinasi YOLOv5 sebagai alat ekstraksi fitur dan KNN sebagai algoritma klasifikasi dalam sistem pemantauan kesehatan ayam [44].

Berdasarkan berbagai penelitian yang telah dilakukan, dapat disimpulkan bahwa algoritma KNN menunjukkan tingkat fleksibilitas yang tinggi serta akurasi yang kompeten dalam mengklasifikasikan kondisi kesehatan ayam, baik melalui data visual maupun data lingkungan. Keunggulan utama KNN terletak pada struktur algoritmanya yang sederhana, efisien dalam hal komputasi, serta mampu bekerja secara optimal pada sistem real-time dengan perangkat berdaya rendah

Dalam penelitian ini, KNN akan diimplementasikan bersama metode deteksi citra menggunakan YOLOv5, di mana fitur visual seperti postur tubuh, pola gerakan, dan warna jengger ayam akan dijadikan input untuk proses klasifikasi. Pendekatan ini diharapkan dapat menghasilkan sistem yang responsif, akurat, dan dapat diterapkan secara efektif dalam lingkungan peternakan unggas modern**.**

2.1.4 *Framework* dan *Tools* yang Digunakan

2.1.4.1 Python, PyTorch, OpenCV

Gambar 4. Aplikasi Python + colab

Python merupakan salah satu bahasa pemrograman yang sangat diminati dalam pengembangan kecerdasan buatan karena memiliki sintaks yang sederhana dan didukung oleh komunitas pengguna yang luas. Bahasa ini tidak hanya banyak digunakan di dunia pendidikan dan penelitian, tetapi juga memiliki berbagai pustaka penting untuk komputasi ilmiah, seperti *NumPy* dan *SciPy*. Dalam bidang pembelajaran mesin, Python menyediakan dukungan untuk pustaka populer seperti *Scikit-learn* dan *PyTorch*, yang memudahkan implementasi algoritma baik secara *supervised* maupun *unsupervised*. Selain itu, Python juga mendukung integrasi dengan pustaka seperti *OpenCV*, sehingga memungkinkan pemrosesan citra dan video secara efisien dan waktu nyata *(real-time)* [45].

Gambar 5. Proses Deteksi Pytorch

PyTorch merupakan *framework deep learning open-source* yang menawarkan fleksibilitas tinggi dan menerapkan pendekatan *"define-by-run"*, yang menjadikannya ideal untuk keperluan riset dan pengembangan prototipe karena kemudahan dalam proses *debugging*. Sementara itu, *OpenCV* adalah pustaka *open-source* yang dirancang untuk aplikasi visi komputer. Dengan *OpenCV*, berbagai tugas seperti pengenalan gerakan tangan, *deteksi objek*, hingga klasifikasi gambar dapat dilakukan, yang sangat penting dalam sistem pembelajaran berbasis bahasa isyarat maupun aplikasi lain dalam pengolahan citra [46].

2.1.4.2 *Google Colab, Jupyter Notebook*

Google Colab merupakan layanan *cloud* gratis dari *Google* yang memungkinkan pengguna mengeksekusi kode *Python* langsung melalui *browser*, dengan dukungan *GPU* dan *TPU* yang mempercepat proses pelatihan model *deep learning*. Salah satu kelebihan utama dari platform ini adalah kemudahannya dalam penggunaan—tidak memerlukan instalasi di perangkat lokal dan terhubung langsung dengan *Google Drive*, sehingga memudahkan penyimpanan dan kolaborasi. *Google Colab* sangat ideal digunakan dalam konteks pembelajaran dan pelatihan, terutama di bidang *kecerdasan buatan* dan *machine learning*, karena memungkinkan pengguna untuk langsung menerapkan konsep teori ke dalam praktik pemrograman [47].

Di sisi lain, *Jupyter Notebook* merupakan perangkat penting dalam mendukung praktik sains terbuka dan pencatatan proses penelitian secara interaktif. Platform ini memungkinkan pengguna untuk menulis kode, menampilkan visualisasi, dan mendokumentasikan penjelasan dalam satu wadah, sehingga sejalan dengan prinsip *FAIR* (*Findable*, *Accessible*, *Interoperable*, *Reusable*). Dalam konteks penelitian, *Jupyter* berperan sebagai media yang memfasilitasi *transparansi*, *reproduksibilitas*, dan kolaborasi ilmiah dengan cara

menyatukan data, skrip, serta hasil analisis dalam satu dokumen yang terintegrasi [48].

2.1.4.3 *LabelIng*

Gambar 7.Proses Labeling

LabelImg merupakan aplikasi anotasi grafis berbasis *Python* yang berfungsi untuk memberi label pada objek dalam sebuah gambar. Alat ini sangat bermanfaat dalam menyusun dataset untuk keperluan deteksi objek, seperti pada model *YOLO*, *SSD*, atau *Faster-RCNN*. Melalui *LabelImg*, pengguna dapat dengan mudah menandai area objek pada citra dan menyimpannya dalam format *XML* (*Pascal VOC*) maupun *TXT* (*YOLO*), yang nantinya digunakan sebagai data untuk melatih model deteksi. *LabelImg* memiliki keunggulan pada antarmuka-nya yang sederhana namun tetap kaya fungsi. Dalam proyek yang membutuhkan anotasi manual dalam jumlah besar, mulai dari ratusan hingga ribuan gambar, alat ini dapat mempercepat pekerjaan berkat dukungan navigasi melalui *keyboard* dan fitur penyimpanan otomatis. *LabelImg* menjadi komponen penting dalam proyek *machine learning* berbasis *computer vision*, terutama untuk model yang membutuhkan data berlabel dengan tingkat akurasi tinggi dalam proses pelatihannya [49].

2.1.4.4 *Scikit-learn*

Scikit-learn merupakan pustaka *Python* yang menyediakan beragam algoritma *machine learning* untuk berbagai keperluan, seperti klasifikasi, regresi, klastering, pemilihan *fitur*, dan evaluasi *model*. Dengan desain yang menekankan kemudahan penggunaan serta antarmuka *API* yang konsisten, *Scikit-learn* menjadi pilihan tepat bahkan bagi pengguna yang bukan ahli di bidang *pembelajaran mesin*. Pustaka ini juga memanfaatkan struktur data dari *NumPy* dan *SciPy*, sehingga dapat dengan mudah diintegrasikan dengan pustaka-pustaka ilmiah *Python* lainnya. *Scikit-learn* juga dikenal karena menjaga kualitas kodenya melalui pengujian unit yang menyeluruh serta dokumentasi yang komprehensif. *Framework* ini memudahkan pengguna untuk melakukan eksperimen dengan

berbagai algoritma secara efisien menggunakan objek *estimator*, teknik *cross-validation*, dan *pipeline*. Selain itu, perbandingan performa model dapat dilakukan dengan mudah menggunakan *GridSearchCV* atau *validasi silang* yang dapat dijalankan secara paralel di beberapa inti prosesor. Berkat lisensi *BSD* yang bersifat permisif, *Scikit-learn* banyak digunakan di dunia akademik maupun industri [50].

2.2 Kesehatan Ayam Petelur

Kondisi kesehatan pada ayam petelur merupakan aspek mendasar dalam menunjang kesuksesan kegiatan peternakan, karena berdampak langsung terhadap jumlah produksi telur, kualitas cangkang, serta durasi masa produksi yang optimal. Berbagai jenis penyakit, baik yang bersifat kronis maupun akut, dapat dengan mudah menyebar di sistem peternakan intensif yang memiliki kepadatan populasi tinggi, sehingga diperlukan pengawasan secara kontinu. Inovasi melalui teknologi citra digital dan *Artificial Intelligence (AI)* menjadi alternatif efektif untuk mengurangi ketergantungan pada metode pemantauan manual. Sebuah *systematic review* mengungkapkan bahwa pemanfaatan algoritma seperti *Convolutional Neural Network (CNN)*, *YOLOv5*, *YOLOv4*, serta model-model *object detection* lainnya mengalami perkembangan yang sangat pesat dalam mengenali perilaku tidak normal pada unggas, dan menunjukkan akurasi deteksi yang sangat tinggi terhadap berbagai parameter Kesehatan [51].

Pemantauan kesehatan ayam berbasis kamera—termasuk analisis terhadap warna jengger (*comb color*)—telah terbukti efektif dalam proses diagnosis tanpa perlu interaksi langsung dengan hewan. Sebuah studi dari Taiwan mengintegrasikan *YOLOv4* dengan sistem kalibrasi warna untuk melacak perubahan warna jengger pada ayam broiler selama masa pertumbuhannya. Pendekatan ini memungkinkan deteksi gejala penyakit secara otomatis melalui sistem *video surveillance* [52].

Seiring dengan kemajuan teknologi *Artificial Intelligence (AI)*, algoritma seperti *YOLO* (*You Only Look Once*) mulai dimanfaatkan untuk mendeteksi kondisi kesehatan ayam melalui citra dari rekaman video. *YOLO* merupakan salah satu pendekatan *deep learning* yang dirancang untuk melakukan *object detection* secara cepat dan presisi. Dengan menggunakan sistem ini, ayam yang menunjukkan tanda-tanda sakit—seperti postur tubuh yang tidak wajar, jengger yang tampak pucat, atau penurunan aktivitas—dapat teridentifikasi secara otomatis melalui tangkapan kamera. Model *YOLOv5* sendiri telah terbukti efektif dalam mengenali ayam sakit maupun mati, dengan nilai *mean Average Precision (mAP)* yang melebihi 97% [53].

Guna meningkatkan ketepatan dalam mengklasifikasikan kondisi kesehatan ayam, hasil deteksi dari *YOLO* dapat digabungkan dengan algoritma *K-Nearest Neighbor (KNN)*. Melalui *KNN*, berbagai fitur visual yang ditangkap oleh kamera—seperti gerakan tubuh, posisi ayam, dan warna jengger (*comb*)—dapat dianalisis

untuk menentukan apakah ayam berada dalam keadaan sehat atau tidak. Integrasi antara *YOLO* dan *KNN* telah terbukti menghasilkan kinerja yang lebih konsisten dalam sistem pemantauan unggas secara *real-time*. Bahkan, penelitian terkini telah mengombinasikan *YOLOv8* dengan metode pelacakan seperti *DeepSORT*, dan menunjukkan tingkat akurasi pelacakan yang mencapai lebih dari 94% [54].

2.2.1 Ciri-ciri Ayam sehat dan Ayam sakit

Berikut ini adalah ciri ayam yang berada dalam kondisi sehat menunjukkan kerakteristik berikut:

1. Memiliki tingkat aktivitas yang tinggi serta menunjukkan perilaku alami seperti makan, minum, berjalan, dan berinteraksi dengan pejantan, yang mencerminkan kondisi kesehatan dan tingkat *well-being* yang baik.
2. Mengeluarkan suara secara normal tanpa adanya indikasi stres; suara terdengar jernih dan tidak disertai batuk, bersin, maupun napas tersengal- sengal.
3. Menunjukkan penampilan fisik yang prima, ditandai dengan bulu yang rapi, jengger yang cerah, serta postur tubuh yang tegak.
4. Menghasilkan feses dengan warna dan tekstur yang normal, sebagai indikasi bahwa sistem pencernaan dan keseimbangan *microbiota* dalam tubuh berfungsi dengan baik [55].

Selanjutnya mengenai dengan ayam yang sedang mengalami gangguan Kesehatan, memperlihatkan gejala-gejala sebagai berikut:

1. Aktivitas menurun drastis; ayam cenderung diam atau berdiri tanpa gerak, terutama saat mengalami gejala penyakit seperti *fowl typhoid* yang disebabkan oleh *Salmonella Gallinarum.*
2. Mengeluarkan vokalisasi yang tidak wajar, misalnya suara serak, batuk, atau tanda distress akibat demam atau infeksi pada saluran pernapasan.
3. Memperlihatkan ciri fisik yang memburuk, seperti bulu tampak kusam, jengger yang terlihat pucat, dan postur tubuh yang menurun.
4. Menghasilkan *feses* yang tidak normal—baik dari segi tekstur maupun warna—yang menjadi indikasi adanya gangguan sistem pencernaan atau infeksi pada saluran usus [56].

2.2.1.1 Ciri visual yang dapat diamati dari kondisi fisik Ayam

Pengamatan secara visual terhadap kondisi ayam petelur merupakan langkah awal yang efektif dalam mengidentifikasi potensi gangguan kesehatan. Dalam sistem peternakan yang telah terotomatisasi, pendekatan ini diperkuat oleh penggunaan kamera digital dan algoritma berbasis *Artificial Intelligence* seperti *K-*

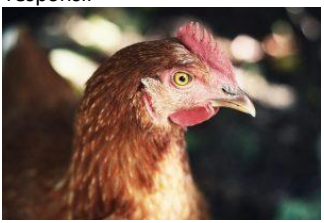
Nearest Neighbor (KNN) serta *YOLOv5*. Teknologi ini memungkinkan sistem untuk mengelompokkan ayam secara otomatis ke dalam dua klasifikasi utama, yakni ayam sehat (*Class* 1) dan ayam sakit (*Class* 2). Proses klasifikasi ini dilakukan dengan menganalisis ciri-ciri visual yang terekam melalui kamera dan diproses secara langsung oleh sistem cerdas [57].

1. Ayam sehat (*class* 1)

   Berikut ini adalah tabel dari ciri-ciri visual ayam petelur yang sehat.

Table 5. Tabel ciri-ciri ayam petelur kondisi sehat

No.	Ciri Visual	Keterangan
1	Postur tubuh tegap dan gerakan aktif 	Mencerminkan kondisi fisik yang baik
2	Jengger merah terang dan tegak 	Menunjukkan aliran darah lancar dan metabolisme normal
3	Bulu rapi dan mengilap 	Menandakan ayam tidak mengalami stres atau gangguan pada kulit

4	Mata terbuka, bening, dan responsif 	Mengindikasikan kondisi sadar penuh dan tidak lemas

2. Ayam sakit *(class2)*
Berikut ini adalah tabel yang menampilkan ciri-ciri visual ayam petelur yang sedang dalam kondisi sakit:

Table 6. Tabel ciri-ciri ayam petelur kondisi sakit

No.	Ciri Visual	Keterangan
1	Posisi tubuh membungkuk atau tidak bergerak lama 	Menunjukkan kelelahan Ekstrem atau kemungkinan infeksi parah
2	Jengger pucat atau menghitam 	Mencerminkan gangguan sirkulasi darah atau gejala anemia

3	Bulu kusam dan rontok 	Bisa menjadi tanda stres, kekurangan nutrisi, atau serangan parasit seperti kutu
4	Aktivitas rendah,terisolasi, enggan makan/minum 	Menandakan kondisi lemah, kurang nafsu makan, atau potensi gangguan kesehatan yang serius

Integrasi kamera & algoritma *KNN*

Sistem klasifikasi berbasis *K-Nearest Neighbor (KNN)* memanfaatkan kamera digital untuk merekam citra ayam secara langsung di dalam kandang. Citra yang diperoleh kemudian dianalisis dengan membandingkan kemiripan visualnya terhadap dataset pelatihan yang telah diberi label sesuai dengan gejala klinis. Setiap citra akan dikategorikan ke dalam *Class* 1 atau *Class* 2 berdasarkan tingkat kemiripan terhadap data sebelumnya. Visualisasi pada gambar yang dilampirkan menggambarkan keluaran dari sistem klasifikasi berbasis *KNN*, dengan rincian:

1. Empat gambar di bagian atas (*Class* 1) memperlihatkan ayam yang berada dalam kondisi sehat.

2. Empat gambar di bagian bawah (*Class* 2) menampilkan ayam yang sedang sakit, disertai ciri visual yang mencolok.

Gambar 8. Tanda gejala klinis [58]

2.2.1.2 Faktor-Faktor yang Memengaruhi Kesehatan Ayam Petelur

Selanjutnya ini ada beberapa factor yang mempengaruhi Kesehatan Ayam Petelur:

1. Kualitas Nutrisi dan air minum
   Pemberian nutrisi yang tepat dan seimbang—terutama kandungan *protein*, *calcium*, *phosphorus*, *vitamin*, dan *minerals*—memegang peran krusial dalam menjaga sistem kekebalan tubuh, kesehatan tulang, serta mutu hasil produksi telur. Ketidakseimbangan atau kekurangan nutrisi dapat menyebabkan terbentuknya cangkang telur yang tipis dan mudah pecah. Selain itu, kualitas air minum, termasuk kandungan *mineral* serta potensi adanya *contaminants*, juga sangat menentukan kelancaran fungsi fisiologis dan proses pencernaan pada ayam [59].

2. Kondisi lingkungan dan udara
   Suhu lingkungan yang terlalu tinggi dapat memicu *heat stress* pada ayam, yang berdampak pada penurunan konsumsi pakan serta mengganggu keseimbangan *acid–base* dalam darah. Kondisi ini secara langsung memengaruhi kesehatan ayam dan kualitas cangkang telur. Di samping itu, konsentrasi gas berbahaya seperti *ammonia* dan *carbon dioxide* (CO_2) yang tinggi di dalam kandang berisiko menimbulkan gangguan pernapasan, kerusakan pada bulu (*feather damage*), serta mendorong perilaku agresif seperti *pecking* [60].

3. Kepadatan populasi & sistem kandang

Tingkat kepadatan yang tinggi dalam kandang dapat memicu persaingan dalam mengakses pakan, menimbulkan stres sosial, serta meningkatkan kemungkinan terjadinya *cannibalism* dan *feather pecking*. Selain itu, penggunaan sistem kandang tertutup seperti *battery cages* cenderung membatasi ruang gerak dan aktivitas fisik ayam, yang berkontribusi pada meningkatnya risiko *osteoporosis* akibat kurangnya pergerakan alami. Di samping itu, rancangan kandang seperti penggunaan *enriched cages*— dan ketersediaan fasilitas pendukung memiliki peranan penting dalam menunjang tingkat kesejahteraan serta kondisi kesehatan unggas secara keseluruhan [61].

4. Manajemen dan kebersihan kandang
   Pengelolaan kebersihan kandang yang tidak memadai, lemahnya sistem *quarantine*, serta keberadaan hama seperti tikus dan kutu dapat memicu berbagai infeksi, termasuk yang disebabkan oleh bakteri *Salmonella* dan *Escherichia coli (E. coli)*. Berdasarkan hasil *review* ilmiah, kegagalan dalam proses *disinfection*, tingginya populasi hama, ketidakteraturan usia ayam dalam satu kandang, serta lemahnya penerapan *biosecurity* merupakan faktor utama yang menyebabkan munculnya kontaminasi patogen di lingkungan peternakan [62].

5. Factor genetic, usia, dan proses *molting*
   Ayam petelur yang telah memasuki usia lanjut cenderung mengalami gangguan tulang seperti *osteoporosis*. Selain itu, usia yang semakin tua turut berkontribusi terhadap penurunan kualitas cangkang telur, perubahan ukuran telur, serta penurunan daya tahan tubuh. Prosedur *forced molting*—yaitu pemaksaan ayam untuk menghentikan produksi dan mengganti bulu—berpotensi melemahkan sistem kekebalan dan meningkatkan risiko infeksi *Salmonella.*

6. Kesejahteraan hewan *(Animal Welfare)*
   Tingkat stres pada ayam dapat dikurangi melalui pengaturan pencahayaan, pemberian rangsangan lingkungan (*enrichment*), dan manajemen kandang yang tepat, sehingga mampu menekan munculnya perilaku agresif seperti *feather pecking*. Berdasarkan kajian ilmiah, unsur seperti pengaturan intensitas cahaya, kontrol kebisingan (*sound management*), manajemen transisi antar fase produksi, serta pelatihan petugas peternakan merupakan

komponen penting dalam menjaga kesejahteraan unggas dan mencegah gangguan perilaku [63].

2.2.2 Dataset Gambar Ayam

Dataset gambar yang dimanfaatkan dalam sistem deteksi kesehatan ayam petelur mencakup beragam jenis citra, termasuk gambar ayam maupun telur, yang memiliki tingkat detail cukup tinggi untuk mendukung proses pelatihan model *YOLOv5* dan *K-Nearest Neighbor (KNN).* Penjelasan berikut merinci secara lengkap setiap jenis data visual yang digunakan dalam sistem ini.

Gambar 9. Gambar Ayam Petelur

2.2.2.1 Jenis dan sumber data yang digunakan

Jenis dataset gambar yang digunakan dalam sistem berisi:

1. Citra ayam dalam kandang baterai
   Dari berbagai sudut pandang Jenis data ini berisi gambar ayam dari tampak depan, samping, dan atas. Tujuannya adalah untuk melatih model agar mampu mengenali karakteristik visual dari berbagai arah. Teknik pengambilan gambar *multi-view* terbukti membantu meningkatkan akurasi proses deteksi objek oleh *YOLO* serta klasifikasi oleh *K-Nearest Neighbor (KNN)* [64].

2. Gambar ayam sehat dan sakit
   Dataset ini mencakup beragam fitur visual seperti warna jengger, postur tubuh, dan kondisi bulu ayam. Proses *annotasi* dilakukan dengan dua kategori utama: *Class* 1 untuk ayam sehat dan *Class* 2 untuk ayam sakit, sehingga sistem dapat membedakan kedua kondisi tersebut secara otomatis [65].

3. Citra telur abnormal (bercangkang tipis atau pecah)
   Gambar telur dengan kondisi tidak normal juga digunakan sebagai indikator tambahan, karena ada kaitan erat antara status kesehatan

ayam dengan kualitas telur. Teknik *machine vision* dan *spectroscopy* tanpa perusakan (*non-destructive*) digunakan untuk mendeteksi telur yang mengalami retak, cangkang tipis, atau bercak darah [66].

4. Data hasil *preprocessing* dari kamera CCTV atau IP camera
   Jenis data ini diperoleh dari kamera otomatis yang dipasang di kandang, kemudian melalui tahapan *preprocessing* seperti *resizing*, konversi ke *grayscale*, dan penambahan *bounding box*. Setelah itu, data diberi label oleh peternak atau pakar, lalu digunakan untuk melatih model *YOLO* dan *KNN* dalam mendukung klasifikasi ayam secara *real-time* [67].

Berikut adalah tabel yang menyajikan sumber data citra ayam petelur berdasarkan jenis dan keterangan:

Table 7. Tabel sumber data citra ayam petelur

No.	Sumber Data	Keterangan
1	Data lapangan milik sendiri	Citra dikumpulkan secara langsung dari kandang ayam petelur lokal yang dipantau selama 24 jam menggunakan kamera CCTV.
2	*Platform Kaggle – Poultry Dataset*	Beberapa dataset terbuka, seperti *"Poultry Disease Classification",* dimanfaatkan untuk melengkapi data penyakit ayam dari berbagai negara.
3	Citra hasil eksperimen klasifikasi dengan *KNN*	Seperti ditampilkan pada Gambar 2, gambar ini memperlihatkan distribusi antara *Class* 1 dan *Class* 2 hasil klasifikasi model *KNN.*

2.2.2.2 Proses pelabelan dan anotasi data

Tahap pelabelan dan anotasi data merupakan langkah mendasar dalam sistem *machine learning* berbasis citra, terutama untuk kebutuhan deteksi serta klasifikasi objek seperti ayam petelur. Tanpa keberadaan label yang tepat dan konsisten, algoritma seperti *YOLOv5* dan *K-Nearest Neighbor (KNN)* tidak akan mampu mempelajari pola secara optimal dari dataset yang disediakan [68].

1. Langkah Anotasi dan Alat yang Digunakan:
   Pelabelan dilakukan dengan menandai objek ayam dalam gambar menggunakan *bounding box*, kemudian menetapkan label kategori yang sesuai. Beberapa alat (*tools*) yang umum dipakai dalam proses ini antara lain:

1. *LabelImg*: digunakan untuk menghasilkan berkas dalam format *XML* (*Pascal VOC*) atau *TXT* (format *YOLO*) dengan anotasi *bounding box* manual.
2. *Roboflow*: sebuah platform *cloud-based* yang menyediakan fitur anotasi otomatis, konversi format, serta augmentasi data.
3. *LabelMe*: perangkat *open-source* yang dikembangkan oleh *MIT*, mendukung anotasi berbasis *polygon* dan menghasilkan file *JSON* sesuai format *COCO*.
4. Label yang diberikan pada objek dalam gambar umumnya menggambarkan kondisi dan perilaku ayam, seperti: ayam sehat, ayam sakit, ayam makan, ayam tidur, ayam menyendiri, dan ayam tidak aktif.

2. Format File yang Digunakan:
   Pemilihan format anotasi disesuaikan dengan jenis algoritma yang digunakan, antara lain:
   1. *YOLO:* menggunakan file berformat *TXT* yang berisi koordinat *bounding box* dengan struktur *class* x, *center* y, *center width height* dalam bentuk *normalized.*
   2. Pascal VOC: menggunakan format *XML* yang berisi informasi posisi
      bounding box serta label kelas objek.
   3. COCO: menggunakan format JSON yang mendukung deteksi banyak objek sekaligus *(multiple objects)* dan anotasi segmentasi per objek *(instance segmentation).*

Anotasi tujuan utama dari proses anotasi adalah memberikan *ground truth* pada setiap citra dalam dataset, yang nantinya dijadikan acuan dalam pelatihan model. Dalam penelitian ini, anotasi dilakukan untuk membedakan apakah ayam berada dalam kondisi sehat atau sakit, serta mencatat perilaku tertentu yang dapat dikenali dari citra kamera. Proses ini juga memungkinkan evaluasi performa model menggunakan metrik seperti *precision*, *recall*, dan *mean Average Precision (mAP)* berdasarkan keberhasilan deteksi terhadap label yang telah ditentukan [69].

Bab 3. Metodologi Penelitian

Untuk proses pengerjaan tugas akhir berjudul "Deteksi dan Prediksi Kesehatan Ayam Menggunakan *YOLOv5* dan *KNN* dilaksanakan melalui serangkaian tahapan yang terstruktur, dimulai dari tahap perancangan, pengumpulan dataset, penerapan *model*, pengujian sistem, hingga penyusunan laporan akhir. Tujuan utama dari pelaksanaan ini adalah untuk membangun sistem yang mampu secara otomatis mengenali tanda-tanda penyakit pada ayam melalui citra digital, serta mengelompokkan kondisinya dengan memanfaatkan algoritma *machine learning*. Diagram di bawah ini menggambarkan alur umum pelaksanaan dari proyek tugas akhir tersebut.

3.1 Gambaran Umum Perancangan Sistem

Perancangan sistem ini diawali dengan penentuan topik tugas akhir yang disesuaikan dengan minat serta urgensi penerapan teknologi di sektor peternakan, khususnya dalam hal pemantauan kesehatan ayam berbasis *Artificial Intelligence* (AI). Ide tersebut muncul dari pengamatan langsung di lapangan, yang menunjukkan kesulitan peternak dalam mengenali gejala penyakit sejak awal akibat kurangnya

tenaga ahli dan belum tersedianya sistem pemantauan otomatis. Oleh karena itu, sistem ini dirancang dengan mengintegrasikan teknologi *object detection* menggunakan algoritma YOLOv5 serta metode klasifikasi *K-Nearest Neighbors (KNN)*. Tinjauan pustaka dilakukan untuk memperkuat landasan teori, mencakup penerapan kamera, *machine learning*, serta algoritma pendeteksi dan pengklasifikasi visual yang relevan pada sektor unggas. Fokus utama sistem adalah mengidentifikasi ciri-ciri visual penyakit seperti perubahan postur, posisi sayap tidak normal, dan kondisi bulu yang tidak sehat.

Setelah perancangan sistem selesai, langkah selanjutnya adalah mengumpulkan citra ayam dalam kondisi sehat dan sakit dari peternakan lokal maupun sumber terbuka, lalu memberi anotasi menggunakan tools seperti *LabelImg* dan *Roboflow*. Data tersebut digunakan untuk melatih model *YOLOv5* dengan ukuran input 416x416 piksel dan parameter seperti *confidence threshold*

0.25 serta IoU 0.5 guna meningkatkan ketepatan deteksi. Hasil deteksi berupa *bounding box* dan label diproses lebih lanjut oleh algoritma KNN untuk klasifikasi status kesehatan ayam, dengan parameter K antara 3 hingga 5 dan pengukuran jarak berbasis Euclidean. Kinerja sistem kemudian dievaluasi melalui metrik seperti *precision*, *recall*, dan akurasi. Jika hasil evaluasi belum mencapai target, dilakukan peningkatan performa melalui penyesuaian parameter atau penambahan data. Setelah sistem mencapai performa yang memadai, disusun laporan akhir dan dilakukan presentasi, dengan tujuan agar sistem dapat langsung dimanfaatkan di lingkungan peternakan secara praktis dan efisien.

3.2 Perancangan Sistem

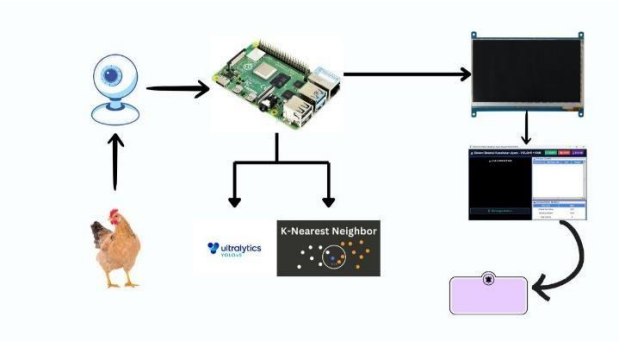
3.2.1 Perancangan Cara Kerja Sistem

Sistem untuk mendeteksi dan memprediksi kesehatan ayam berbasis *YOLO-KNN* ini dirancang guna mendukung peternak dalam memantau kondisi ayam secara otomatis melalui analisis citra visual. Perancangannya memadukan teknologi *object detection* menggunakan *YOLOv5* dan metode *machine learning* berupa *K-Nearest Neighbors (KNN)*, yang diintegrasikan dengan sistem penyimpanan berbasis *cloud* seperti *Firebase* atau *database* digital lainnya.

Pengguna hanya perlu mengakses sistem ini melalui aplikasi di *smartphone* atau melalui *web-based dashboard*. Gambar ayam yang ditangkap oleh kamera akan diproses secara otomatis oleh sistem—dimulai dari deteksi visual, pengklasifikasian kondisi kesehatan, hingga pengiriman *notifikasi* hasil prediksi. Seluruh rangkaian proses ini berlangsung secara otomatis, efisien, dan mendukung pengoperasian secara *real-time*.

3.2.2 Diagram Blok Sistem

Berdasarkan perancangan cara kerja sistem yang telah dijelaskan pada Subbab 3.2.1 berikut merupakan tahapan gambar diagram blok dan penjelasan cara kerja sistem deteksi dan prediksi kesehatan ayam.



Gambar 10. Diagram Blok Sistem

1. Pengambilan Gambar Ayam / web cam
   Perangkat kamera seperti CCTV atau modul *IoT* digunakan untuk secara otomatis menangkap gambar ayam di dalam kandang. Proses pengambilan ini dapat dilakukan secara periodik berdasarkan waktu tertentu, atau melalui sistem pemicu berbasis gerakan (*motion detection*).

2. Penyimpanan Citra ke Basis Data

Gambar-gambar yang berhasil ditangkap akan disimpan ke dalam *database* untuk keperluan dokumentasi, sekaligus sebagai bahan pelatihan dan pengujian dalam pengembangan model kecerdasan buatan.

3. Labeling Dataset menggunakan *Roboflow*
   Sebelum model YOLOv5 dapat digunakan untuk mendeteksi kondisi kesehatan ayam, citra yang telah dikumpulkan harus dianotasi terlebih dahulu. Proses anotasi dilakukan menggunakan platform *Roboflow*, yaitu platform berbasis web yang dirancang untuk memudahkan proses pemberian label *(labeling)* pada gambar dalam proyek *computer vision*. Dalam penelitian ini, Roboflow digunakan untuk memberi label pada citra ayam, baik yang diperoleh dari proses *data collecting* secara langsung maupun dari dataset sekunder seperti Kaggle.

Table 8. Tahapan proses labeling menggunakan Roboflow

No	Langkah	Deskripsi
1	Upload Gambar	Unggah gambar ayam (sehat & sakit) ke Roboflow
2	Labeling	Gambar bounding box dan beri label sesuai kondisi ayam
3	Export	Ekspor dataset dalam format YOLOv5 (.txt label + gambar)
4	Integrasi	Dataset digunakan dalam pelatihan model YOLOv5

4. Proses Deteksi Awal dengan *YOLOv5*
   Citra yang telah disimpan dianalisis menggunakan algoritma *YOLOv5* (*You Only Look Once*). Sistem ini bertugas mendeteksi keberadaan objek ayam serta mengidentifikasi bagian tubuh tertentu yang mengindikasikan kelainan atau gejala penyakit. Beberapa ciri visual yang dikenali antara lain:
   a) Warna jengger yang tampak lesu
   b) Gerakan kepala yang tidak normal (*head rolling*)
   c) Kondisi paru-paru yang terlihat pucat
   d) Kaki yang mengalami pembengkakan
   e) Munculnya lendir (*snot*) di mata atau hidung.

Gejala tersebut mengarah pada dugaan adanya penyakit tertentu, seperti:
a) Gumboro

b) Kolera dan *Coryza*
c) Pullorum (berak kapur)
d) Newcastle Disease (Tetelo/ND)

5. Ekstraksi Ciri Visual dan Klasifikasi dengan *KNN*
Fitur-fitur hasil dari deteksi, seperti label objek, koordinat *bounding box*, dan tingkat kepercayaan (*confidence score*), akan diolah menjadi input vector untuk tahap klasifikasi menggunakan algoritma *K-Nearest Neighbors (KNN)*. Proses klasifikasi dilakukan dengan menghitung jarak (biasanya menggunakan *Euclidean distance*) antara data baru dengan data latih, untuk menentukan apakah ayam tergolong normal atau abnormal.

Table 9. Gejala visual ayam penyakit

No.	Gejala Visual	Kemungkinan Penyakit yang Terkait
1	Munculnya lendir *(snot)* di mata atau hidung	Infeksi saluran pernapasan seperti *Newcastle Disease (ND)*
2	Sayap tampak terkulai atau ayam terlihat pasif	Indikasi penyakit *Gumboro*
3	Gerakan kepala abnormal *(head rolling)* atau kaki bengkak	Gejala yang mengarah ke Marek atau *Chronic Respiratory Disease (CRD)*

Table 10. Tabel input Algoritma K-Nearest Neighbor (KNN)

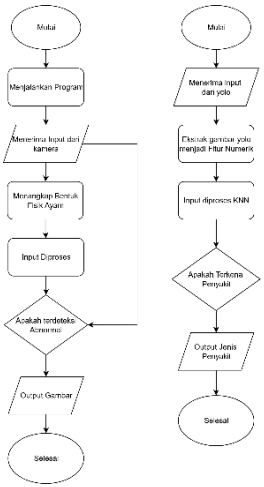
No.	Komponen Input untuk ***Algoritma K-Nearest Neighbor (KNN)***	Keterangan
1	Dataset pelatihan	Berisi contoh ayam dalam kondisi sehat dan sakit sebagai data referensi
2	Nilai K	Jumlah *n_neighbors* terdekat yang digunakan untuk klasifikasi (misalnya 3 atau 5)
3	Metode pengukuran jarak	Menggunakan *Euclidean distance* untuk menentukan kesamaan antar data

6. Notifikasi Hasil Prediksi
Setelah klasifikasi dilakukan, hasil prediksi dikirim ke perangkat pengguna dalam bentuk notifikasi melalui aplikasi *smartphone*. Sistem akan menampilkan kondisi ayam berdasarkan ID masing-masing,

sekaligus menyarankan tindakan pencegahan atau penanganan apabila ditemukan potensi penyakit.

7. Penyimpanan Riwayat ke *Firebase*
   Seluruh hasil deteksi dan klasifikasi, termasuk informasi waktu (*timestamp*) dan identitas ayam, akan disimpan secara otomatis ke *cloud database* seperti *Firebase*. Data ini berguna untuk memantau riwayat kesehatan ayam dari waktu ke waktu, sehingga peternak dapat melakukan pengawasan yang lebih efektif secara historis.

3.2.3 Flowchart Arsitektur Sistem YOLOv5 dan KNN



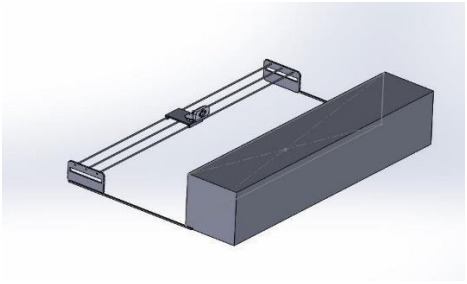
Gambar 11. Flowchart Deteksi dan Klasifikasi Penyakit Ayam.

Perancangan sistem ini terbagi menjadi dua blok utama yang bekerja secara berkesinambungan, yaitu blok akuisisi data dan deteksi awal, serta blok ekstraksi fitur dan klasifikasi penyakit.

Tahap Deteksi dan Pra-proses (YOLOv5) Proses dimulai dengan inisialisasi program yang secara otomatis mengaktifkan perangkat kamera sebagai sensor input utama. Sistem melakukan pengambilan data visual secara *real-time* untuk memantau kondisi fisik ayam di area pantauan. Di sini, sistem melakukan pemindaian untuk menemukan objek ayam dan menentukan apakah terdapat indikasi "abnormal" pada bentuk fisiknya. Jika sistem mendeteksi adanya kejanggalan, gambar objek tersebut akan diisolasi dan dikeluarkan sebagai *output* gambar (hasil *crop* dari *bounding box* YOLO) yang menjadi data mentah untuk tahap selanjutnya.

Tahap Ekstraksi Fitur dan Klasifikasi (KNN) Memasuki tahap kedua, data gambar yang dihasilkan oleh YOLO diterima oleh sistem untuk dilakukan proses ekstraksi fitur. Pada tahap ini, informasi visual dari gambar dikonversi menjadi fitur numerik (seperti nilai RGB, tekstur, atau bentuk) agar dapat diproses secara matematis. Data numerik tersebut kemudian dimasukkan ke dalam algoritma *K-Nearest Neighbor* (KNN) untuk dibandingkan dengan data latih yang sudah ada. KNN akan melakukan klasifikasi untuk menentukan status kesehatan ayam melalui analisis kemiripan data. Jika ayam terklasifikasi sakit, sistem akan menampilkan jenis penyakitnya secara spesifik sebagai

3.2.4 Perancangan Sistem Mekanikal



Gambar 12. Desain Mekanikal

Pada tahap perancangan sistem mekanikal, fokus utama adalah bagaimana menempatkan dan mengatur perangkat keras secara efisien agar proses pengambilan gambar ayam berlangsung dengan konsisten dan akurat. Hal ini sangat penting karena kualitas gambar yang diperoleh akan berpengaruh langsung terhadap kinerja *object detection* menggunakan *YOLOv5*. Oleh karena itu, beberapa faktor teknis seperti kondisi kandang, pencahayaan, stabilitas perangkat, dan sudut pengambilan gambar harus diperhitungkan dengan akurat.

1. Komponen Perangkat Keras yang digunakan:

Table 11. Tabel komponen perangkat keras

No.	Komponen	Deskripsi
1	Kamera	Menggunakan dua opsi: web cam untuk pengawasan stabil dan luas.
2	Dudukan Kamera	Menggunakan tripod untuk instalasi sementara, dan tiang penyangga permanen untuk pemasangan jangka panjang di sekitar kandang.

3	Pelindung Kamera	Pelindung transparan dari akrilik atau plastik digunakan di lingkungan luar atau lembap untuk menjaga kamera tetap aman dan berfungsi baik.

2. Strategi Penempatan Kamera:
   Penemapatan kamera dilakukan dengan mempertimbangankan titik pandang yang mampu menagkap area tubuh ayam yang paling relevan untuk deteksi gejala.

Table 12. Tabel pengaturan pemasangan kamera

No.	Parameter	Pengaturan	Tujuan
1	Jarak pemasangan kamera	Sekitar 0,5 hingga 1 meter dari ayam	Memberikan hasil gambar yang jelas dan fokus
2	Sudut pengambilan gambar	Antara 45° hingga 90°, disesuaikan dengan model Kandang dan perilaku ayam	Memastikan area penting seperti kepala, mata, sayap, dan kaki terekam

3.2.5 Perancangan Sistem Elektrikal

Desain sistem elektrikal dalam penelitian ini difokuskan untuk menjamin agar semua komponen elektronik yang terlibat dalam proses deteksi dan prediksi kondisi kesehatan ayam dapat beroperasi secara efisien dan terintegrasi dengan baik. Setiap perangkat saling terkoneksi mulai dari alat pengambil citra (kamera), unit pengolah data (mikrokontroler), hingga pengirim hasil klasifikasi ke s*erver* atau aplikasi pengguna. Selain itu, perancangan sistem ini juga memperhitungkan keterbatasan daya dan jaringan yang umum ditemukan di lingkungan peternakan.

1. Sistem Penyedia Daya
   Aspek kelistrikan merupakan fondasi penting untuk menjaga agar system tetap aktif. Oleh karena itu, ada sumber energi yang disiapkan untuk menyesuaikan dengan kondisi pertenakan:

Table 13.Tabel mengenai penggunaan adaptor

No.	Komponen	Spesifikasi	Fungsi
1	Adaptor DC	5V–12V	Menyediakan daya bagi mikrokontroler, kamera, dan sensor lainnya

2. Komponen Elektrikal Utama
   Ada beberapa komponen utama dalam sistem ini adalah:

Table 14. Tabel komponen utama

No.	Komponen	Fungsi Utama	Keterangan
1	Raspberry Pi	Pemrosesan utama: menjalankan YOLOv untuk deteksi objek dan KNN Untuk klasifikasi kesehatan ayam.	Mendukung pustaka Python, menyimpan data lokal, mengirim hasil ke dashboard, dan bisa terhubung ke cloud.
2	Icd 7 inch	Media tampilan untuk menampilkan hasil deteksi dan klasifikasi sistem.	Digunakan sebagai antarmuka visual (user interface) yang menampilkan citra, status kondisi ayam, dan informasi sistem secara real-time.
3	Webcam	Perangkat akuisisi citra sebagai sumber data visual ayam.	Berfungsi menangkap citra atau video ayam secara langsung untuk diproses oleh sistem deteksi berbasis YOLOv5.
4	adaptor	Penyedia catu daya untuk mendukung operasional sistem.	Digunakan untuk memasok tegangan dan arus listrik yang stabil ke Raspberry Pi, LCD, dan perangkat pendukung lainnya.

3.2.6 Perancangan Sistem Notifikasi Telegram

Sistem notifikasi Telegram dirancang sebagai sarana penyampaian informasi dari sistem kepada pengguna secara otomatis. Notifikasi ini digunakan untuk mengirimkan informasi terkait proses deteksi dan klasifikasi kondisi kesehatan ayam yang telah dilakukan oleh sistem.

Pengiriman notifikasi dilakukan melalui Telegram Bot yang terhubung menggunakan Telegram API. Bot tersebut berfungsi sebagai media komunikasi antara sistem deteksi berbasis YOLOv5 dan sistem klasifikasi berbasis KNN dengan pengguna. Setiap keluaran hasil pemrosesan data akan diproses terlebih dahulu untuk menentukan format pesan yang akan dikirimkan melalui bot.

Mekanisme pengiriman notifikasi dimulai setelah sistem menyelesaikan proses deteksi objek ayam dan penentuan kondisi kesehatannya. Informasi yang telah diproses kemudian dikirimkan secara real-time tanpa memerlukan interaksi langsung dari pengguna. Dengan mekanisme ini, sistem mampu memberikan

informasi secara cepat dan terstruktur.

Pada Bab III, pembahasan sistem notifikasi Telegram difokuskan pada perancangan dan mekanisme kerja sistem. Implementasi sistem serta contoh notifikasi yang dihasilkan akan dibahas lebih lanjut pada Bab IV sebagai bagian dari hasil pengujian sistem.

3.3 Data Collecting

Tahap *data collecting* merupakan proses pengumpulan serta persiapan data citra ayam yang digunakan sebagai input pada sistem deteksi dan klasifikasi kondisi kesehatan ayam. Data yang dikumpulkan terdiri dari dua jenis, yaitu data yang digunakan untuk pelatihan model deteksi objek menggunakan YOLO dan data yang digunakan dalam proses klasifikasi menggunakan metode K-Nearest Neighbor (KNN).

3.2.1 Data Collecting YOLO

Data YOLO berupa sekumpulan citra ayam yang digunakan dalam proses deteksi objek serta identifikasi kondisi ayam berdasarkan karakteristik visual. Data ini dikelompokkan ke dalam dua kategori utama, yaitu ayam dengan kondisi normal dan ayam dengan kondisi abnormal.

1. Karakteristik Ayam Normal
   Ayam yang berada dalam kondisi normal memiliki beberapa ciri visual sebagai berikut:
   A. Postur tubuh terlihat tegap dan seimbang
   B. Kondisi bulu tampak rapi, bersih, dan tidak kusam
   C. Tidak ditemukan luka, bercak, maupun kelainan pada bagian kulit dan wajah
   D. Aktivitas ayam terlihat aktif serta responsive

   Karakteristik tersebut dijadikan sebagai acuan dalam proses pelabelancitra ayam normal pada sistem YOLO.

2. Karakteristik Ayam Abnormal
   Ayam dengan kondisi abnormal menunjukkan adanya perbedaan atau penyimpangan dari kondisi normal, antara lain:
   A. Bulu terlihat kusam, rontok, atau menggumpal
   B. Terdapat bercak, luka, atau perubahan warna pada kulit dan wajah
   C. Postur tubuh tampak tidak normal atau terlihat lemah
   D. Perilaku ayam cenderung pasif atau kurang aktif

   Pada tahap YOLO, ayam abnormal tidak diklasifikasikan berdasarkan jenis penyakit tertentu, melainkan hanya dideteksi sebagai ayam dengan kondisi tidak normal.

3. Pembagian Dataset YOLO

Dataset citra YOLO dibagi ke dalam tiga bagian, yaitu:

A. **Training set (70%)**, digunakan untuk melatih model YOLO agar mampu mengenali pola visual ayam
B. **Validation set (20%)**, digunakan untuk mengevaluasi performa model selama proses pelatihan
C. **Test set (10%)**, digunakan untuk menguji kemampuan model dalam mendeteksi objek pada data yang belum pernah digunakan sebelumnya

Pembagian dataset ini dilakukan untuk mengurangi risiko *overfitting* serta memastikan model memiliki kemampuan generalisasi yang baik. Skema pembagian dataset ditampilkan dalam bentuk diagram segitiga dengan proporsi 70%, 20%, dan 10%.

Gambar 13. Pembagian Dataset untuk Pelatihan Model YOLO

3.2.2 Data Collecting KNN

Pada penelitian ini, data kesehatan ayam diklasifikasikan ke dalam beberapa kategori penyakit berdasarkan ciri-ciri visual dan kondisi fisik yang umum ditemukan pada ayam. Setiap kategori penyakit memiliki karakteristik khas yang dapat diamati secara visual, sehingga dapat dijadikan acuan dalam proses pengambilan data citra dan pelabelan dataset untuk metode K-Nearest Neighbors (KNN).

Penentuan kategori penyakit dilakukan berdasarkan gejala yang terlihat secara fisik pada ayam, seperti perubahan bentuk kepala, kondisi jengger, gangguan pernapasan, serta munculnya kelainan pada kulit. Informasi mengenai ciri-ciri penyakit ini digunakan sebagai dasar dalam proses pelabelan data citra sebelum dilakukan ekstraksi fitur dan proses klasifikasi lebih lanjut.

Dengan adanya deskripsi ciri penyakit yang jelas, proses pengumpulan data dan penyusunan dataset menjadi lebih terarah, sehingga dapat meningkatkan konsistensi antara kondisi visual ayam dengan label kelas yang digunakan dalam sistem klasifikasi.

Table 15. Kelas Penyakit Ayam pada Dataset KNN

No	Nama Penyakit	Ciri – ciri penyakit
1	Chronic Respiratory Disease (CRD)	Penyakit pernapasan kronis yang ditandai dengan pembengkakan pada bagian kepala, keluarnya lendir dari hidung, serta kondisi ayam yang tampak lemas.
2	Chicken Favus	Penyakit akibat infeksi jamur yang ditandai dengan munculnya kerak berwarna putih atau kekuningan pada jengger, wajah, serta bagian kulit ayam lainnya.
3	Croza	Penyakit pernapasan akut yang ditandai dengan pembengkakan pada wajah, keluarnya cairan dari hidung, serta gangguan pernapasan.
4	Newcastle Disease	Penyakit virus yang menyerang sistem pernapasan dan saraf, dengan gejala seperti leher terpuntir, kelumpuhan, serta penurunan aktivitas ayam.
5	Avian Influenza	Penyakit virus yang bersifat sangat menular, ditandai dengan pembengkakan pada kepala, perubahan warna jengger menjadi kebiruan, serta tingkat kematian yang tinggi.
6	Fowlpox	Penyakit virus yang ditandai dengan munculnya bintik atau benjolan menyerupai cacar pada kulit ayam, terutama di sekitar wajah dan jengger.

Data citra dari masing-masing kelas penyakit tersebut digunakan sebagai data latih dan data uji pada metode KNN untuk mengklasifikasikan kondisi kesehatan ayam berdasarkan tingkat kedekatan jarak fitur.

3.4 Perancangan Dataset

Agar sistem deteksi dan klasifikasi kondisi kesehatan ayam dapat bekerja secara optimal, proses pelatihan model memerlukan pembagian dataset yang terstruktur. Pengelompokan data ini memegang peranan penting karena akan berpengaruh langsung terhadap kualitas hasil pelatihan serta tingkat akurasi model yang dibangun. Dataset dibagi ke dalam tiga kategori utama, yakni data *training*, *validation*, dan *testing*.

Data *training* dimanfaatkan untuk membangun kemampuan model YOLOv5 dalam mendeteksi objek ayam serta untuk melatih algoritma KNN dalam melakukan klasifikasi berdasarkan fitur visual yang ditangkap.

Selama pelatihan, data *validation* digunakan untuk mengevaluasi kinerja model secara berkala. Proses ini bertujuan untuk menghindari overfitting dan membantu dalam pengaturan parameter model agar dapat beradaptasi dengan baik terhadap data yang belum pernah ditemukan sebelumnya.

Sementara itu, data *testing* berfungsi sebagai alat ukur akhir untuk menilai kemampuan generalisasi model setelah seluruh proses pelatihan selesai,

khususnya terhadap data yang benar-benar baru dan tidak digunakan dalam pelatihan maupun validasi.

Table 16. Struktur Pembagian Dataset

Jenis data	Persentase %	Tujuan
Training	70%	Melatih model YOLOv5 dan KNN
Validation	20 %	Validasi performa selama training
Testing	10 %	Uji akhir model setelah pelatihan

Proses pembagian ini dilakukan secara proporsional menggunakan platform *Roboflow*, yang memudahkan pengelolaan dataset serta menyediakan fitur pelabelan data secara efisien.

3.4.1 Dataset Deteksi Ayam (YOLOv5)

Dataset deteksi ayam digunakan pada proses pendeteksian objek ayam menggunakan metode You Only Look Once version 5 (YOLOv5). Dataset ini terdiri dari kumpulan citra ayam yang diambil dari berbagai kondisi lingkungan kandang, dengan variasi posisi, sudut pandang, dan jarak pengambilan gambar. Dataset tersebut digunakan untuk melatih model agar mampu mendeteksi keberadaan ayam pada citra secara akurat.

Setiap citra pada dataset deteksi ayam telah melalui proses anotasi dengan memberikan bounding box pada objek ayam. Proses anotasi dilakukan menggunakan format yang sesuai dengan kebutuhan YOLOv5, yaitu dengan menentukan koordinat bounding box serta label kelas objek. Dataset kemudian dibagi menjadi data latih (training data), data validasi (validation data), dan data uji (testing data) untuk mendukung proses pelatihan dan pengujian model deteksi.

Pembagian dataset dilakukan untuk memastikan bahwa model YOLOv5 mampu melakukan deteksi secara optimal pada data yang belum pernah dilatih sebelumnya. Dataset deteksi ayam ini berperan sebagai tahap awal dalam sistem, karena hasil deteksi ayam selanjutnya digunakan sebagai masukan pada proses klasifikasi kesehatan ayam menggunakan metode K-Nearest Neighbors (KNN).

3.4.2 Dataset Klasifikasi Kesehatan Ayam (KNN)

Dataset yang digunakan pada tahap klasifikasi kesehatan ayam dengan metode K-Nearest Neighbors (KNN) disusun berdasarkan hasil ekstraksi fitur citra ayam. Proses ekstraksi fitur dilakukan setelah objek ayam berhasil dideteksi menggunakan metode YOLOv5. Setiap citra yang telah terdeteksi kemudian diolah untuk memperoleh karakteristik visual yang direpresentasikan dalam bentuk nilai numerik.

Fitur-fitur yang digunakan dalam dataset ini mencakup informasi warna, tingkat keabuan, distribusi histogram, serta karakteristik tepi citra. Pada komponen warna, ekstraksi dilakukan pada kanal B, G, dan R yang masing-masing menghasilkan nilai rata-rata (mean) dan simpangan baku (standard deviation).

Selain itu, histogram warna pada setiap kanal dibagi ke dalam beberapa bin untuk menggambarkan sebaran intensitas warna secara lebih detail.

Selain fitur warna, dilakukan pula ekstraksi fitur citra grayscale yang meliputi nilai rata-rata, simpangan baku, nilai minimum, dan nilai maksimum intensitas keabuan. Histogram grayscale juga digunakan untuk menangkap pola distribusi intensitas cahaya pada citra ayam. Untuk memperkuat karakteristik visual, fitur tepi citra diekstraksi guna merepresentasikan tekstur dan bentuk objek ayam, yang ditunjukkan melalui nilai mean, standard deviation, nilai minimum, maksimum, serta median tepi.

Secara keseluruhan, dataset klasifikasi ini terdiri dari 55 fitur numerik yang merepresentasikan karakteristik visual citra ayam. Struktur dataset dan daftar fitur yang digunakan dalam proses klasifikasi KNN disajikan pada Tabel 3.X. Nilai numerik dari masing-masing fitur akan dibahas lebih lanjut pada Bab IV sebagai bagian dari hasil implementasi dan pengujian sistem.

Table 17. Rancangan Dataset Fitur Klasifikasi Kesehatan Ayam

No	Nama Fitur	Nilai
1	B_mean	
2	B_std	
3	G_mean	
4	G_std	
5	R_mean	
6	R_std	
7	hist_B_0	
8	hist_B_1	
9	hist_B_2	
10	hist_B_3	
11	hist_B_4	
12	hist_B_5	
13	hist_B_6	
14	hist_B_7	
15	hist_G_0	
16	hist_G_1	
17	hist_G_2	
18	hist_G_3	
19	hist_G_4	
20	hist_G_5	
21	hist_G_6	
22	hist_G_7	
23	hist_R_0	

24	hist_R_1	
25	hist_R_2	
26	hist_R_3	
27	hist_R_4	
28	hist_R_5	
29	hist_R_6	
30	hist_R_7	
31	gray_mean	
32	gray_std	
33	gray_min	
34	gray_max	
35	hist_gray_0	
36	hist_gray_1	
37	hist_gray_2	
38	hist_gray_3	
39	hist_gray_4	
40	hist_gray_5	
41	hist_gray_6	
42	hist_gray_7	
43	hist_gray_8	
44	hist_gray_9	
45	hist_gray_10	
46	hist_gray_11	
47	hist_gray_12	
48	hist_gray_13	
49	hist_gray_14	
50	hist_gray_15	
51	edge_mean	
52	edge_std	
53	edge_min	
54	edge_max	
55	edge_median	

3.5 Metode Evaluasi Model

Setelah pelatihan model YOLOv5 dan KNN selesai, langkah berikutnya adalah melakukan evaluasi terhadap kinerjanya. Tujuannya adalah untuk mengetahui sejauh mana efektivitas model dalam mendeteksi dan mengklasifikasikan kondisi ayam berdasarkan data yang tersedia. Evaluasi dilakukan dengan sejumlah metrik umum yang sesuai dengan jenis algoritma masing-masing.

3.5.1 Metode Evaluasi Model YOLOv5

YOLOv5 merupakan model deteksi objek berbasis CNN yang mampu mengidentifikasi ayam dan menghasilkan *bounding box* pada gambar. Evaluasi terhadap performanya dilakukan menggunakan beberapa metrik sebagai berikut:

Table 18. Metrik Evaluasi Model YOLOv5

Metrik	**Penjelasan**
Precision	Rasio deteksi benar terhadap semua prediksi positif
Recall	Rasio deteksi benar terhadap semua objek sebenarnya
mAP (*mean Average Precision*)	Ukuran umum akurasi deteksi bounding box
IoU (*Intersection over Union*)	Ukuran overlap antara prediksi dan ground truth

Penggunaan metrik precision dan recall membantu mengukur seberapa akurat model dalam membedakan ayam sehat dan sakit, sedangkan mAP dan IoU menilai ketepatan posisi prediksi objek.

3.5.2 Metode Evaluasi Model KNN

Model KNN bertugas untuk melakukan klasifikasi berdasarkan fitur visual dari hasil deteksi YOLOv5, seperti perbedaan warna bulu, bentuk tubuh, dan perilaku ayam. Evaluasi model ini menggunakan metrik evaluasi umum dalam klasifikasi data sebagai berikut:

Table 19. Metrik Evaluasi Model KNN

Metrik	**Penjelasan**
Confusion Matrix	Matriks yang menampilkan prediksi benar/salah tiap kelas
Accuracy	Persentase total prediksi yang benar
Precision	Prediksi positif yang benar dibagi total prediksi positif
Recall	Jumlah positif yang terdeteksi benar dibanding total positif sebenarnya
F1-Score	Harmonik rata-rata dari Precision dan Recall

Evaluasi dengan metrik-metrik tersebut membantu memastikan bahwa model tidak hanya akurat, tetapi juga andal dan seimbang dalam mengenali berbagai kondisi kesehatan ayam, terlebih jika terdapat ketimpangan jumlah data antara ayam sehat dan sakit.

Penjelasan *Confusion Matrix*

Confusion Matrix adalah sebuah tabel evaluatif yang berfungsi untuk mengukur kinerja suatu model klasifikasi dengan membandingkan antara label yang sebenarnya (*ground truth*) dan hasil prediksi dari model. Dalam kasus klasifikasi biner seperti yang digunakan pada penelitian ini—yakni mengklasifikasikan ayam menjadi dua kategori: "Normal" dan "AbNormal"—*Confusion Matrix* memiliki format matriks berukuran 2x2 seperti tabel berikut.

Table 20. Empat Komponen Utama Confusion Matrix

	Prediksi: AbNormal	**Prediksi: Normal**
Aktual: Normal	True Positive (TP)	False Negative (FN)
Aktual: AbNormal	False Positive (FP)	True Negative (TN)

Table 21. Penjelasan Komponen

Komponen	**Penjelasan**
TP (True Positive)	Model memprediksi ayam sakit, dan faktanya memang sakit.
TN (True Negative)	Model memprediksi ayam sehat, dan faktanya memang sehat.
FP (False Positive)	Model memprediksi ayam sakit, padahal sebenarnya sehat.
FN (False Negative)	Model memprediksi ayam sehat, padahal sebenarnya sakit.

Table 22. Rumus Perhitungan Metrik Evaluasi

Metrik	**Rumus**
Accuaracy (Akurasi)	(TP + TN) / (TP + TN + FP + FN) =. . %
Precision (presisi)	TP / (TP + FP) =. . %
Recall	TP / (TP + FN) =. . %
F1-Score	2 × (Precision × Recall) / (Precision + Recall) =. . %

3.6 Alat dan Bahan

Tabel berikut menyajikan daftar alat dan bahan yang dirancang sesuai dengan kebutuhan tugas akhir berjudul Deteksi & Prediksi Kesehatan Ayam dengan *YOLOv5* dan *KNN*, lengkap dengan estimasi harga sebagai acuan dalam pelaksanaan tugas akhir.

Table 23. Estimasi biaya

No.	Alat/bahan	Harga Satuan (Rp.)	Jumlah	Total (Rp.)	Keterangan
1	USB *Webcam* HD	300.000	2	600.000	Pribadi
2	*Raspberry Pi* 4 (4GB RAM)	1.100.000	1	1.100.000	Pribadi
3	MicroSD 32GB + *Reader*	80.00	1	80.000	Pribadi
4	*Adaptor DC 5V- 12V*	50.000	1	50.000	Pribadi
5	*Laptop*	-	1	-	Pribadi
	Total :			1.830.000	Pribadi

3.7 Pengujian Sistem

Proses pengujian dilakukan untuk menilai sejauh mana sistem deteksi dan prediksi kesehatan ayam berbasis YOLOv5 dan K-Nearest Neighbors (KNN) dapat bekerja sesuai dengan tujuan perancangan. Pengujian pada penelitian ini difokuskan pada aspek metodologis, yaitu bagaimana sistem diuji, parameter apa saja yang digunakan, serta skenario evaluasi yang diterapkan untuk mengukur kinerja sistem secara menyeluruh. Pada tahap ini belum dibahas hasil numerik maupun analisis performa, karena seluruh hasil pengujian disajikan dan dibahas secara rinci pada Bab IV.

Pengujian mencakup beberapa tahapan utama, antara lain pengujian dataset, pengujian algoritma deteksi objek menggunakan YOLOv5, pengujian algoritma klasifikasi kesehatan ayam menggunakan KNN, serta pengujian sistem terintegrasi YOLOv5–KNN. Setiap tahapan pengujian dirancang untuk memastikan bahwa sistem mampu melakukan deteksi ayam, mengidentifikasi kondisi normal dan abnormal, serta mengklasifikasikan jenis penyakit ayam berdasarkan fitur visual yang diperoleh dari citra.

Selain itu, pengujian juga mempertimbangkan aspek keterpaduan sistem secara keseluruhan, mulai dari proses pengambilan citra menggunakan kamera, pemrosesan data citra oleh algoritma YOLOv5, hingga proses klasifikasi penyakit menggunakan KNN. Evaluasi kinerja sistem dirancang menggunakan beberapa metrik standar, seperti confusion matrix, precision, recall, F1-score, accuracy, dan confidence score. Seluruh metrik tersebut digunakan sebagai acuan evaluasi untuk menilai keandalan dan stabilitas sistem, serta menjadi dasar dalam analisis hasil pengujian pada bab selanjutnya.

3.7.1 Pengujian Dataset Deteksi Ayam (YOLOv5)

Pengujian dataset deteksi ayam dilakukan untuk memastikan bahwa data yang dirancang memiliki pemisahan kelas yang jelas antara ayam dengan kondisi normal dan ayam dengan kondisi abnormal. Dataset ini digunakan sebagai dasar dalam proses training dan testing pada model YOLOv5, sehingga perancangannya harus mampu merepresentasikan kondisi kesehatan ayam secara visual dengan baik.

Penyusunan dataset dilakukan dengan mengelompokkan data berdasarkan sumber data serta jenis kondisi kesehatan ayam. Pada Bab III, pembahasan difokuskan pada perencanaan dan struktur dataset**,** meliputi konsep pengelompokan data dan kriteria kondisi ayam yang digunakan dalam sistem deteksi. Oleh karena itu, pada tahap ini tidak disajikan jumlah data secara rinci.

Adapun jumlah data aktual serta hasil pengujian dataset akan dijelaskan secara lengkap pada Bab IV sebagai bagian dari analisis dan pembahasan, setelah proses implementasi dan pengujian model YOLOv5 dilakukan.

3.7.2 Pengujian Dataset Klasifikasi Kesehatan Ayam (KNN)

Pengujian dataset klasifikasi kesehatan ayam dengan algoritma K-Nearest Neighbors (KNN) dilakukan untuk merancang sistem pengelompokan kondisi kesehatan ayam berdasarkan fitur visual yang diperoleh dari hasil proses deteksi sebelumnya. Dataset ini digunakan sebagai dasar dalam proses training dan testing guna memastikan algoritma KNN mampu melakukan klasifikasi kondisi kesehatan ayam secara tepat dan konsisten.

Dataset diklasifikasikan ke dalam beberapa kelas kondisi kesehatan ayam sesuai dengan jenis gangguan yang diamati. Perancangan pembagian kelas ini bertujuan untuk meningkatkan kemampuan algoritma KNN dalam mengukur tingkat kemiripan antar data berdasarkan perhitungan jarak fitur. Dengan pembagian kelas yang jelas, proses klasifikasi diharapkan dapat menghasilkan keputusan yang lebih akurat.

Pada Bab III, pembahasan difokuskan pada struktur dan perencanaan dataset klasifikasi**,** meliputi konsep pembagian kelas kondisi kesehatan ayam serta peran dataset sebagai data latih dan data uji dalam algoritma KNN. Oleh karena itu, jumlah data dan hasil pengujian klasifikasi tidak disajikan pada bab ini**.** Adapun rincian jumlah dataset serta hasil pengujian klasifikasi KNN disajikan secara lengkap pada Bab IV sebagai bagian dari analisis dan pembahasan hasil penelitian.

3.7.3 Pengujian Deteksi Ayam Menggunakan YOLOv5

Pengujian deteksi ayam menggunakan model YOLOv5 dilakukan untuk merancang proses evaluasi kemampuan model dalam mengenali objek ayam serta membedakan kondisi ayam normal dan abnormal. Tahap pengujian ini difokuskan pada perancangan metode evaluasi terhadap hasil deteksi berdasarkan data yang telah melalui proses pelatihan sebelumnya.

Metode evaluasi yang digunakan dalam penelitian ini adalah confusion matrix,

yang berfungsi sebagai alat ukur performa klasifikasi model dengan membandingkan label aktual dan hasil prediksi. Pada Bab III, confusion matrix dijelaskan sebagai konsep dan rancangan evaluasi yang digunakan untuk menganalisis kinerja model YOLOv5 dalam mendeteksi dan mengklasifikasikan kondisi kesehatan ayam.

Oleh karena itu, nilai evaluasi kuantitatif seperti jumlah prediksi benar dan salah tidak disajikan pada bab ini. Adapun hasil pengujian deteksi beserta nilai metrik evaluasi, seperti accuracy, precision, recall, dan F1-score, akan disajikan dan dibahas secara rinci pada Bab IV sebagai bagian dari analisis hasil penelitian.

3.7.4 Pengujian Evaluasi Metrik YOLOv5

Pengujian evaluasi metrik pada model YOLOv5 dilakukan untuk merancang metode penilaian performa model dalam mendeteksi dan mengklasifikasikan kondisi kesehatan ayam. Parameter evaluasi yang digunakan meliputi precision, recall, F1-score, dan confidence, yang berfungsi sebagai indikator tingkat ketepatan dan keandalan model dalam membedakan kondisi ayam normal dan abnormal.

Pada Bab III, pembahasan difokuskan pada penyusunan kerangka evaluasi dan penjelasan parameter metrik yang digunakan untuk menilai kinerja model YOLOv5. Oleh karena itu, nilai hasil pengujian tidak disajikan pada bab ini**.** Adapun hasil evaluasi metrik serta analisis performa model YOLOv5 disajikan dan dibahas secara rinci pada Bab IV berdasarkan hasil deteksi yang diperoleh dari proses pengujian sistem.

3.7.5 Pengujian Klasifikasi Kesehatan Ayam Menggunakan KNN

Pengujian klasifikasi kesehatan ayam menggunakan algoritma K-Nearest Neighbors (KNN) dirancang untuk mengevaluasi kemampuan model dalam mengelompokkan kondisi kesehatan ayam berdasarkan tingkat kemiripan fitur visual yang diekstraksi dari data citra. Pengujian ini bertujuan untuk memastikan bahwa algoritma KNN mampu melakukan klasifikasi penyakit ayam secara sistematis dan konsisten.

Pada Bab III, pembahasan difokuskan pada perancangan skema evaluasi serta penentuan metrik penilaian yang digunakan untuk mengukur kinerja model KNN, seperti precision, recall, F1-score, dan confidence. Oleh karena itu, hasil nilai evaluasi dan ringkasan performa model tidak disajikan pada bab ini**.** Adapun **hasil** klasifikasi serta analisis performa model KNN disajikan dan dibahas secara rinci pada Bab IV sebagai bagian dari hasil dan pembahasan penelitian

3.7.6 Pengujian Interpretasi Metrik KNN

Pengujian interpretasi metrik pada algoritma K-Nearest Neighbors (KNN) dilakukan untuk merancang proses pembandingan antara nilai hasil evaluasi klasifikasi dengan target performa yang telah ditetapkan sebelumnya. Tahap ini bertujuan sebagai acuan konseptual dalam menilai tingkat keberhasilan sistem

klasifikasi kesehatan ayam yang dikembangkan.

Pada Bab III, pembahasan difokuskan pada penyusunan indikator evaluasi serta kerangka penilaian kinerja sistem**,** tanpa menyajikan nilai hasil pengujian maupun status capaian. Adapun penentuan status capaian serta interpretasi akhir terhadap hasil evaluasi metrik KNN dilakukan dan dibahas secara rinci pada Bab IV berdasarkan nilai metrik yang diperoleh dari proses pengujian sistem.

3.7.7 Pengujian Sistem Terintegrasi YOLOv5–KNN

Pengujian sistem terintegrasi YOLOv5–KNN dirancang untuk mengevaluasi kinerja sistem secara menyeluruh, mulai dari proses deteksi objek ayam menggunakan model YOLOv5 hingga proses klasifikasi penyakit ayam menggunakan algoritma K-Nearest Neighbors (KNN). Pengujian ini bertujuan untuk memastikan bahwa seluruh alur kerja sistem terintegrasi berjalan sesuai dengan rancangan dan mampu menghasilkan keluaran yang konsisten.

Pada Bab III, pembahasan difokuskan pada perancangan skema evaluasi sistem terintegrasi**,** termasuk penggunaan confusion matrix sebagai metode evaluasi untuk menilai kesesuaian antara label aktual dan hasil prediksi pada setiap kelas penyakit ayam. Oleh karena itu, hasil pengujian dan nilai evaluasi tidak disajikan pada bab ini**.** Adapun hasil pengujian serta analisis performa sistem YOLOv5–KNN secara menyeluruh disajikan dan dibahas secara rinci pada Bab IV sebagai bagian dari hasil dan pembahasan penelitian.

3.7.8 Pengujian Evaluasi Metrik Sistem Gabungan YOLOv5–KNN

Pengujian evaluasi metrik pada sistem gabungan YOLOv5–KNN dirancang untuk menyusun indikator penilaian kinerja sistem secara keseluruhan, mulai dari proses deteksi objek ayam hingga klasifikasi kondisi kesehatan ayam. Evaluasi dilakukan menggunakan parameter precision, recall, F1-score, dan confidence untuk menilai tingkat ketepatan dan keandalan sistem dalam membedakan setiap kategori kondisi kesehatan ayam.

Pada Bab III, pembahasan difokuskan pada perancangan kerangka evaluasi metrik yang digunakan untuk menilai kinerja sistem gabungan YOLOv5–KNN, tanpa menyajikan nilai hasil pengujian. Adapun nilai evaluasi metrik serta analisis kinerja sistem secara menyeluruh disajikan dan dibahas secara rinci pada Bab IV sebagai bagian dari hasil dan pembahasan penelitian.

Bab 4. Hasil dan Pembahasan

Bab ini menyajikan hasil pengujian serta pembahasan dari sistem deteksi dan prediksi kesehatan ayam petelur berbasis YOLOv5 dan *K-Nearest Neighbors* (KNN). Pengujian dilakukan untuk mengevaluasi kinerja sistem secara menyeluruh, mulai dari tahap pengumpulan dataset, proses deteksi objek ayam, klasifikasi kondisi kesehatan, hingga evaluasi akurasi sistem terintegrasi. Hasil yang diperoleh dianalisis untuk mengetahui tingkat keberhasilan sistem serta keterbatasan yang masih terdapat dalam penelitian ini.

4.1 Gambaran Umum Sistem Penelitian

Subbab ini membahas gambaran umum pelaksanaan pengujian dan analisis sistem deteksi serta klasifikasi kesehatan ayam yang dikembangkan menggunakan metode YOLOv5 dan *K-Nearest Neighbor* (KNN). Pengujian dilakukan untuk mengevaluasi kemampuan sistem dalam mendeteksi objek ayam serta mengklasifikasikan kondisi kesehatannya berdasarkan ciri visual. Sistem diuji menggunakan dataset citra ayam yang diperoleh dari berbagai kondisi pencahayaan dan posisi, dengan tujuan mensimulasikan kondisi lingkungan peternakan yang sebenarnya.

4.2 Hasil Persiapan dan Pengolahan Dataset

4.2.1 Pengumpulan dan Sumber Dataset

Dataset merupakan komponen penting dalam pengembangan sistem berbasis machine learning. Pada penelitian ini, dataset disusun sesuai dengan fungsi masing-masing model, yaitu dataset untuk deteksi menggunakan YOLOv5 dan dataset untuk klasifikasi penyakit menggunakan KNN.

Gambar 14. Visualisasi Citra Penyakit Ayam dari Sumber Dataset Penelitian

Dataset YOLOv5 digunakan untuk mendeteksi ayam sekaligus menentukan kondisi kesehatannya ke dalam dua kelas, yaitu *Normal* (ayam sehat) dan *Abnormal* (ayam sakit). Total *dataset YOLOv5* yang digunakan berjumlah 4.258

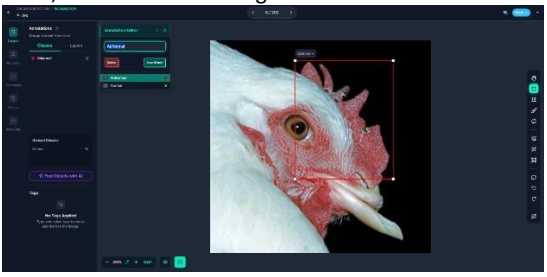
citra, yang terdiri dari 2.317 citra kelas *Abnormal* dan 1.941 citra kelas *Normal*. Data diperoleh dari pengambilan langsung di lapangan serta dari sumber publik, kemudian diseleksi untuk memastikan kualitas citra yang layak digunakan dalam proses *training*.

Setiap citra dianotasi menggunakan platform *Roboflow* dengan memberikan *bounding box* pada objek ayam agar posisi dan ukuran objek dapat dikenali dengan baik oleh model *YOLOv5*. Proses anotasi dilakukan secara manual untuk menjaga konsistensi label serta meminimalkan kesalahan sejak tahap awal.

Sementara itu, *dataset KNN* digunakan untuk melakukan klasifikasi lanjutan terhadap ayam yang terdeteksi *abnormal*. *Dataset* ini terdiri dari enam kategori penyakit, yaitu *Avian Influenza*, *CDR (Chronic Respiratory Disease)*, *Chicken Favus*, *Croza*, *Fowl Pox*, dan *Newcastle Disease*. *Dataset KNN* dibentuk dari hasil ekstraksi fitur visual citra ayam yang telah dideteksi oleh *YOLOv5*, seperti warna jengger, tekstur bulu, kondisi kulit, dan postur tubuh ayam. Pendekatan dua tahap ini membuat sistem bekerja lebih efisien, karena proses klasifikasi penyakit hanya dilakukan pada ayam yang telah terdeteksi dalam kondisi *abnormal* oleh *YOLOv5*.

4.2.2 Proses Pelabelan Data Menggunakan *Roboflow*

Proses anotasi dilakukan menggunakan Roboflow yang menyediakan fasilitas pelabelan citra secara interaktif. Setiap gambar diberi bounding box yang mengelilingi objek ayam secara presisi agar area pembelajaran model terfokus pada objek utama, bukan latar belakang.



Gambar 15. Antarmuka Roboflow pada Proses Anotasi Bounding Box Objek Ayam

Label pada *YOLOv5* dibagi menjadi dua kelas, yaitu *Normal* dan *Abnormal*. Penamaan label dibuat konsisten tanpa spasi atau karakter khusus agar sesuai dengan format standar *YOLO*. Pelabelan dilakukan secara manual untuk menjaga kualitas anotasi. Selain anotasi, *Roboflow* juga digunakan untuk melakukan augmentasi data seperti *rotasi*, *flipping*, dan penyesuaian pencahayaan guna meningkatkan variasi data pelatihan serta membantu model menjadi lebih *robust* terhadap kondisi lingkungan yang berbeda.

4.2.3 Pembagian Dataset *Training, Validation,* dan *Testing*

Gambar 16. Distribusi Train, Valid, dan Test Set

Setelah proses anotasi citra selesai, *dataset* dibagi ke dalam tiga *subset*, yaitu *training*, *validation*, dan *testing*, dengan perbandingan masing-masing sebesar 70%, 20%, dan 10%. Berdasarkan hasil pembagian *dataset* pada sistem, dari total 3.051 gambar diperoleh 2.141 gambar sebagai data *training*, 611 gambar sebagai data *validation*, dan 299 gambar sebagai data *testing*. Pembagian ini bertujuan untuk memastikan model memiliki data yang cukup untuk proses pembelajaran, evaluasi selama pelatihan, serta pengujian kemampuan generalisasi pada data yang belum pernah dilihat sebelumnya.

Dataset yang telah dibagi kemudian diproses menggunakan platform *Roboflow* melalui tahap *processing* dan *augmentation*. Pada tahap *processing*, dilakukan *auto-orient* untuk memastikan orientasi citra tidak terbalik serta *resize* citra menjadi ukuran 640 × 640 piksel, sesuai dengan standar *input* model *YOLOv5*. Selanjutnya, pada tahap *augmentation*, diterapkan teknik *horizontal flip* dengan tujuan meningkatkan kemampuan model dalam mengenali objek pada orientasi kiri dan kanan.

Setelah seluruh tahapan selesai, *dataset* diunduh dalam format *YOLOv5 PyTorch* dengan opsi *show download code*, sehingga memudahkan proses pelatihan model menggunakan *Google Colab*. *Dataset* yang diunduh telah tersusun secara terstruktur sesuai dengan kebutuhan pelatihan. Proses pelatihan model dilakukan menggunakan arsitektur *YOLOv5s*, yang dipilih karena memiliki keseimbangan antara akurasi dan kompleksitas model untuk kebutuhan deteksi secara *realtime*. Parameter pelatihan yang digunakan meliputi ukuran *input* citra 640 × 640 piksel, *batch size* 32, dan 150 *epoch*. Model terbaik dari hasil pelatihan disimpan dalam file dengan ekstensi *best.pt* sebagai model dengan performa terbaik selama proses pelatihan.

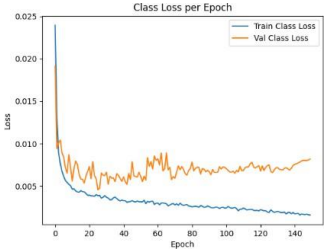
4.3 Hasil Pelatihan Model YOLOv5

4.3.1 Grafik *Training Loss* dan *Validation Loss*

Subbab ini membahas hasil pelatihan model YOLOv5 yang ditunjukkan melalui grafik nilai training loss dan validation loss. Parameter loss yang dianalisis meliputi class loss, object loss, dan box loss untuk mengevaluasi proses pembelajaran

model, tingkat konvergensi, serta kemampuan generalisasi model terhadap data validasi.

1. Grafik Class Loss

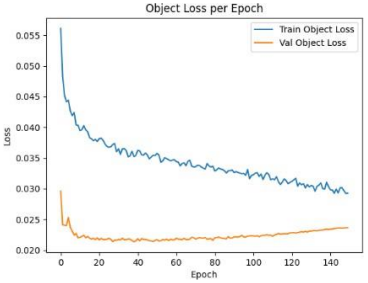


Gambar 17. Kurva Loss Pelatihan dan Validasi

Grafik ini menunjukkan perubahan nilai *class loss* pada data *training* dan data *validation* selama proses *training*. Pada awal *epoch*, nilai *class loss* baik pada data *training* maupun *validation* mengalami penurunan yang cukup tajam, yang menandakan bahwa model dengan cepat mempelajari pola dasar dalam mengklasifikasikan kelas objek. Seiring bertambahnya *epoch*, *class loss* pada data *training* terus menurun secara bertahap hingga mencapai nilai yang rendah, menunjukkan peningkatan kemampuan model dalam membedakan kelas objek secara akurat.

Sebaliknya, nilai *class loss* pada data *validation* cenderung berfluktuasi dan relatif stabil pada nilai yang lebih tinggi dibandingkan data *training*, dengan sedikit kecenderungan meningkat pada *epoch* akhir. Kondisi ini mengindikasikan bahwa meskipun model menunjukkan performa klasifikasi yang baik pada data *training*, terdapat indikasi *overfitting* ringan terhadap data *training*. Namun demikian, perbedaan nilai *class loss* antara data *training* dan *validation* masih berada dalam batas yang dapat diterima, sehingga model tetap memiliki kemampuan generalisasi yang cukup baik dalam melakukan klasifikasi kelas.

2. Grafik Object Loss

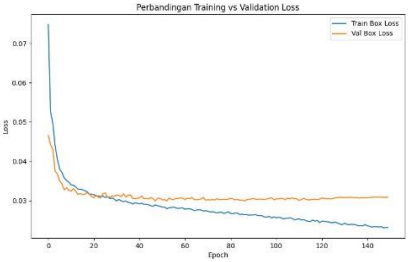


Gambar 18. Grafik Object Loss

Gambar tersebut memperlihatkan grafik *object loss* pada data *training* dan *validation* terhadap *epoch*. Nilai *object loss* pada data *training* menunjukkan tren menurun secara konsisten seiring bertambahnya *epoch*, yang menandakan bahwa model semakin baik dalam mengenali keberadaan objek selama proses *training*.

Pada sisi lain, nilai *object loss* pada data *validation* mengalami penurunan pada awal proses *training* dan kemudian cenderung stabil dengan sedikit peningkatan pada *epoch* akhir. Kondisi ini mengindikasikan bahwa model telah belajar dengan baik dari data *training*, namun mulai menunjukkan gejala *overfitting* ringan, di mana peningkatan performa pada data *training* tidak sepenuhnya diikuti oleh data *validation*. Meskipun demikian, perbedaan nilai *loss* antara data *training* dan *validation* masih berada dalam batas yang wajar, sehingga model tetap memiliki kemampuan generalisasi yang cukup baik.

3. Grafik *Box Loss*



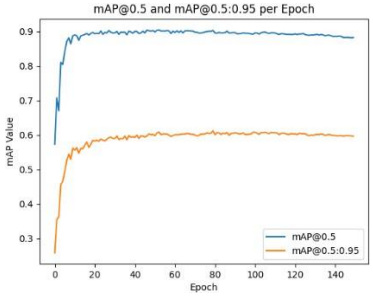
Gambar 19. Perbandingan Loss Koordinat Bounding Box

Gambar tersebut menunjukkan perbandingan nilai *box loss* antara data *training* dan *validation*. Nilai *box loss* pada data *training* menurun secara signifikan

dari awal hingga akhir *epoch*, yang menandakan bahwa model semakin akurat dalam memprediksi posisi *bounding box* objek.

Sementara itu, nilai *box loss* pada data *validation* mengalami penurunan pada awal proses *training*, kemudian relatif stabil dan sedikit meningkat pada *epoch* selanjutnya. Kondisi ini menunjukkan bahwa kemampuan generalisasi model terhadap data *validation* tergolong baik, meskipun terdapat indikasi *overfitting* ringan pada *epoch* akhir. Secara keseluruhan, grafik ini menunjukkan bahwa model telah berhasil mempelajari pola lokasi objek dengan baik dan mencapai kondisi pelatihan yang stabil.

4.3.2 Grafik Nilai *mAP@50* dan *mAP@50:95*



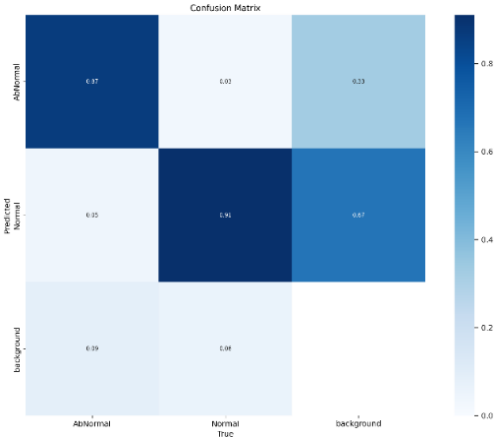
Gambar 20. Grafik mAP Value

Grafik tersebut menunjukkan perubahan nilai *mAP@0.5* dan *mAP@0.5:0.95* seiring bertambahnya jumlah *epoch* selama proses *training*. Nilai *mAP@0.5* meningkat tajam pada tahap awal pelatihan dan kemudian cenderung stabil pada nilai yang tinggi hingga akhir *epoch*, yang menandakan bahwa model dengan cepat mencapai kemampuan deteksi objek yang baik pada batas *IoU* 0.5 serta mampu mempertahankan kinerjanya secara konsisten.

Di sisi lain, nilai *mAP@0.5:0.95* juga mengalami peningkatan yang cukup signifikan pada awal proses *training*, namun berada pada nilai yang lebih rendah dibandingkan *mAP@0.5* karena kriteria evaluasi yang lebih ketat. Setelah mencapai nilai maksimum pada *epoch* menengah, performa *mAP@0.5:0.95* relatif stabil dengan sedikit fluktuasi hingga akhir pelatihan.

Secara umum, pola yang ditunjukkan pada grafik ini mengindikasikan bahwa model berhasil belajar secara efektif, mencapai kondisi konvergensi, serta memiliki performa deteksi objek yang konsisten dan optimal pada data evaluasi.

4.3.3 Confusion Matrix Deteksi Ayam



Gambar 21. Confusion Matrix Model

Confusion matrix pada gambar tersebut menggambarkan tingkat ketepatan model dalam mengklasifikasikan objek ke dalam tiga kelas, yaitu *Abnormal*, *Normal*, dan *background*. Hasil evaluasi menunjukkan bahwa model memiliki kemampuan yang baik dalam mengenali kelas *Abnormal*, yang ditunjukkan oleh tingginya proporsi data *Abnormal* yang berhasil diprediksi secara tepat. Hal ini mengindikasikan bahwa model cukup efektif dalam mendeteksi kondisi yang tidak normal.

Pada kelas *Normal*, model juga menunjukkan performa yang baik dengan tingkat prediksi benar yang relatif tinggi. Namun demikian, masih terdapat sebagian kecil data *Normal* yang keliru diklasifikasikan ke kelas lain. Kesalahan ini mengindikasikan adanya tumpang tindih karakteristik antara kelas *Normal* dengan kelas lainnya, sehingga memengaruhi hasil prediksi.

Untuk kelas *background*, terlihat bahwa model belum sepenuhnya mampu membedakan latar belakang dengan objek secara sempurna. Beberapa data *background* masih terdeteksi sebagai objek *Normal* maupun *Abnormal*, yang menunjukkan adanya kesalahan deteksi berupa *false positive*. Meskipun demikian, secara keseluruhan nilai yang dominan pada diagonal utama *confusion matrix* menunjukkan bahwa model telah memiliki kinerja klasifikasi yang cukup baik dan dapat digunakan untuk membedakan kondisi objek secara efektif.

Untuk memperkuat analisis visual tersebut, evaluasi kinerja model juga dianalisis secara kuantitatif menggunakan confusion matrix numerik yang disajikan pada Tabel 22.

Table 24. Hasil Confusion Matrix YOLOv5

			Prediksi	
		normal	abnormal	none
	normal	25	0	0
Aktual	abnormal	0	24	1
	none	0	0	0

Tabel ini menyajikan *confusion matrix* yang digunakan untuk melakukan evaluasi kinerja klasifikasi model *YOLOv5* secara lebih mendalam. *Confusion matrix* menggambarkan keterkaitan antara kelas aktual yang ditampilkan pada sumbu vertikal dengan kelas hasil prediksi pada sumbu horizontal. Nilai yang berada pada diagonal utama menunjukkan jumlah data yang berhasil diklasifikasikan dengan benar oleh model.

Analisis *Confusion Matrix* :

1. **Kategori normal**: Seluruh 25 data ayam dengan kondisi normal berhasil dikenali dan diprediksi secara tepat sebagai normal, tanpa ditemukan kesalahan klasifikasi.
2. **Kategori abnormal**: Dari total 25 data ayam abnormal, sebanyak 24 data dapat diprediksi dengan benar, sementara 1 data lainnya keliru diklasifikasikan sebagai *none*, yang termasuk dalam kasus *false negative*.
3. **Kategori none**: Tidak terdapat data pengujian pada kategori ini, sehingga seluruh nilai pada kategori tersebut bernilai nol.

Berdasarkan hasil *confusion matrix*, diperoleh sebanyak 49 prediksi yang sesuai dari total 50 data uji. Dengan demikian, tingkat akurasi model *YOLOv5* yang dihasilkan mencapai sebesar 98%.

4.3.4 Evaluasi Metrik Klasifikasi YOLOv5

Table 25. Hasil Evaluasi Metrik YOLOv5

Kategori	Precision	Recall	F1-Score	Rata-Rata Confidence
Normal	1	1	1	0,88
Abnormal	1	0,96	0,98	0,80
Rata-Rata	1	0,98	0,99	0,84

Tabel tersebut menyajikan hasil evaluasi kinerja model YOLOv5 dalam

mendeteksi serta mengklasifikasikan kondisi kesehatan ayam secara real-time. Proses pengujian dilakukan sebanyak 50 kali uji yang mencakup tiga kategori, yaitu normal, abnormal, dan none. Evaluasi performa model dilakukan dengan menggunakan metrik precision, recall, F1-score, serta nilai rata-rata confidence sebagai indikator tingkat kepercayaan model dalam menghasilkan deteksi.

1. **Presisi (Precision)**
   Precision digunakan untuk mengukur tingkat ketepatan model dalam memprediksi suatu kategori tertentu. Metrik ini dihitung menggunakan persamaan berikut: **Precision = TP / (TP + FP)**
   Sebagai contoh, perhitungan precision untuk kategori normal adalah sebagai berikut: **Precision = 25 / (25 + 0) = 25 / 25 = 1,0**

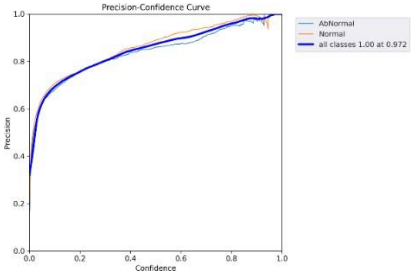
2. **Recall (Sensitivity)**
   Recall menunjukkan kemampuan model dalam mendeteksi seluruh data yang benar-benar termasuk ke dalam suatu kategori. Perhitungan recall dilakukan dengan rumus: **Recall = TP / (TP + FN)**
   Contoh perhitungan recall pada kategori abnormal adalah:
   Recall = 24 / (24 + 1) = 24 / 25 = 0,96

3. **F1-Score**
   F1-score merupakan nilai rata-rata harmonik antara precision dan recall, yang digunakan untuk menilai keseimbangan antara ketepatan dan kelengkapan prediksi model. Nilai F1-score dihitung menggunakan persamaan berikut:
   F1-score ***= 2 × (*****Precision** ***×*** **Recall*****) / (*****Precision** ***+*** **Recall*****)***
   Sebagai contoh, perhitungan F1-score untuk kategori abnormal adalah:
   F1-score = 2 × (1,0 × 0,96) / (1,0 + 0,96) = 2 × 0,96 / 1,96 = 0,98

4.3.5 Analisis Performa YOLOv5 Berdasarkan Confidence Threshold

Selain menggunakan confusion matrix, performa model YOLOv5 juga dievaluasi menggunakan metrik precision, recall, dan F1-score yang dianalisis berdasarkan variasi nilai confidence threshold. Evaluasi ini bertujuan untuk mengetahui tingkat ketepatan prediksi, kelengkapan deteksi objek, serta keseimbangan performa model dalam mendeteksi kelas Normal dan Abnormal.

1. Grafik Precision terhadap Confidence

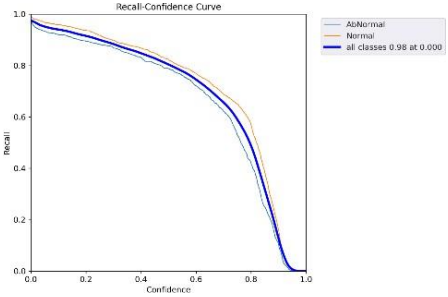


Gambar 22. Grafik Presisi terhadap Kepercayaan Model

Pada kurva Precision–Confidence, nilai precision meningkat seiring bertambahnya confidence threshold. Hal ini menunjukkan bahwa semakin tinggi ambang kepercayaan yang digunakan, maka prediksi model menjadi semakin akurat karena jumlah prediksi salah (false positive) berkurang.

Baik kelas Abnormal maupun Normal memperlihatkan pola peningkatan precision yang serupa. Secara keseluruhan, model mencapai nilai precision maksimum pada tingkat confidence yang sangat tinggi, yang mengindikasikan bahwa model memiliki tingkat keandalan prediksi yang baik ketika hanya mempertimbangkan deteksi dengan tingkat kepercayaan tinggi.

2. Grafik Recall terhadap Confidence



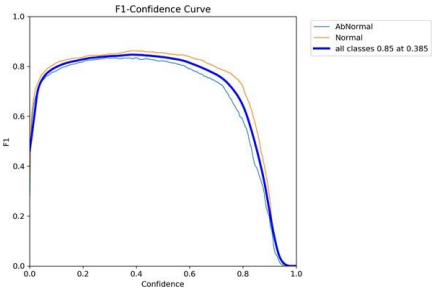
Gambar 23. Kurva Recall-Confidence

Pada kurva Recall–Confidence, nilai recall cenderung menurun seiring meningkatnya confidence threshold. Kondisi ini menandakan bahwa semakin ketat ambang kepercayaan yang diterapkan, maka semakin banyak objek yang tidak terdeteksi oleh model.

Pada nilai confidence yang rendah, recall berada pada tingkat yang tinggi karena hampir seluruh objek berhasil terdeteksi. Namun, nilai recall menurun

secara signifikan ketika confidence mendekati 1, yang menunjukkan bahwa hanya prediksi dengan tingkat kepercayaan sangat tinggi yang dipertahankan oleh model, sehingga jumlah deteksi yang lolos menjadi lebih sedikit.

3. Grafik F1-Score terhadap Confidence



Gambar 24. Kurva F1–Confidence

Sementara itu, pada kurva F1–Confidence terlihat adanya keseimbangan antara nilai precision dan recall. Nilai F1-score meningkat pada tingkat confidence rendah hingga menengah, kemudian menurun kembali pada confidence yang tinggi.

Puncak nilai F1-score menunjukkan titik ambang kepercayaan (confidence threshold) yang paling optimal, di mana model memiliki keseimbangan terbaik antara ketepatan prediksi dan kelengkapan deteksi. Secara keseluruhan, grafik ini mengindikasikan bahwa pemilihan confidence threshold yang tepat sangat penting untuk memperoleh performa model yang optimal dalam proses deteksi objek. Berdasarkan hasil evaluasi melalui grafik Precision–Confidence, Recall–Confidence, dan F1–Confidence, ringkasan nilai performa model YOLOv5 untuk setiap kelas disajikan dalam bentuk tabel.

Table 26. Hasil Evaluasi Performa Model

Class	Precision (P)	Recall (R)	mAP@0.5	mAP@0.5:0.95
Normal	0.857	0.869	0.903	0.613
Abnormal	0.837	0.833	0.875	0.589
All	0.847	0.851	0.931	0.636

Dataset pada penelitian ini dibagi menjadi data training dan data testing sebelum proses pelatihan model dilakukan. Pembagian dataset bertujuan untuk memastikan bahwa evaluasi kinerja model dilakukan secara objektif menggunakan data yang tidak terlibat dalam proses training. Dua skenario pembagian dataset digunakan, yaitu rasio 50:50 dan 50:90, guna menguji performa serta kemampuan generalisasi model terhadap variasi jumlah data uji.

4.3.6 Visualisasi Hasil Deteksi Objek

Gambar 25. Visualisasi Hasil Deteksi Model

Berdasarkan hasil pengujian sistem deteksi menggunakan algoritma YOLO, terlihat bahwa model mampu mengidentifikasi dan mengklasifikasikan kondisi ayam ke dalam dua kelas, yaitu Normal dan Abnormal, dengan tingkat kepercayaan yang bervariasi. Ayam yang terdeteksi sebagai Normal umumnya berada dalam posisi tubuh yang wajar, seperti berdiri atau bergerak secara stabil. Sebaliknya, ayam yang diklasifikasikan sebagai Abnormal menunjukkan kondisi fisik yang tidak normal, seperti terbaring, posisi tubuh terbalik, atau postur yang tidak lazim.

Setiap objek ayam berhasil ditandai dengan bounding box beserta nilai confidence, yang merepresentasikan tingkat keyakinan model terhadap hasil klasifikasi. Variasi nilai confidence dipengaruhi oleh perbedaan sudut pengambilan gambar, kondisi pencahayaan, serta karakteristik fisik ayam itu sendiri.

Secara keseluruhan, hasil pengujian ini menunjukkan bahwa sistem deteksi yang dikembangkan mampu membedakan kondisi ayam Normal dan Abnormal secara visual dengan cukup baik, sehingga dapat digunakan sebagai dasar dalam pemantauan kesehatan ayam secara otomatis.

4.4 Hasil Klasifikasi Kesehatan ayam Menggunakan KNN

Klasifikasi kesehatan ayam pada penelitian ini dilakukan menggunakan algoritma K-Nearest Neighbors (KNN) berdasarkan fitur visual yang diperoleh dari hasil deteksi objek ayam menggunakan YOLOv5. Proses klasifikasi ini bertujuan untuk menentukan kondisi kesehatan ayam berdasarkan tingkat kemiripan fitur citra dengan data training yang telah dikumpulkan sebelumnya.

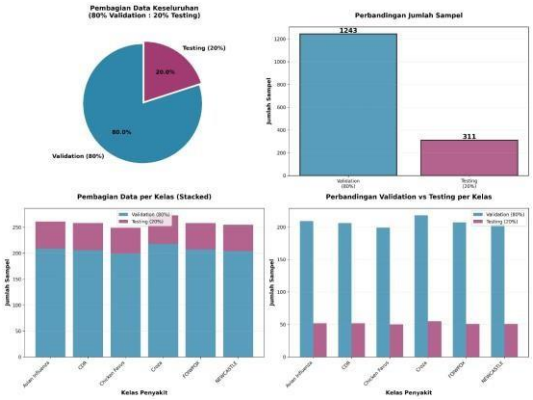
4.4.1 Dataset Fitur Klasifikasi Kesehatan Ayam

No	Nama Fitur	Nilai	Penyakit CDR
1	B_mean	96.841797	
2	B_std	65.009545	
3	G_mean	119.843567	
4	G_std	78.566753	
5	R_mean	161.402588	
6	R_std	90.228176	
7	hist_B_0	25.360107	
8	hist_B_1	14.880371	
9	hist_B_2	9.393311	
10	hist_B_3	13.446045	
11	hist_B_4	9.539795	
12	hist_B_5	24.34082	
13	hist_B_6	2.7771	
14	hist_B_7	0.262451	
15	hist_G_0	17.895508	
16	hist_G_1	20.495605	
17	hist_G_2	3.326416	
18	hist_G_3	6.256104	
19	hist_G_4	12.17041	
20	hist_G_5	10.418701	

Gambar 26. Dataset Fitur Hasil Ekstraksi Citra Ayam Klasifikasi KNN

Analisis terhadap dataset fitur hasil ekstraksi citra dilakukan pada tahap ini sebagai parameter input utama bagi algoritma *K-Nearest Neighbors* (KNN) dalam proses klasifikasi penyakit ayam. Hasil ekstraksi menunjukkan bahwa setiap kategori penyakit memiliki karakteristik statistik warna yang unik; sebagai contoh, *Avian Influenza* ditandai dengan nilai *Red mean* yang dominan (184,87) yang merepresentasikan warna kemerahan pada jengger, sedangkan *Newcastle Disease* menunjukkan nilai RGB yang lebih tinggi dan seimbang (rata-rata 153) yang mengindikasikan kenampakan visual pucat. Selain itu, tekstur pada *Chicken Favus* memiliki tingkat kontras yang tajam dengan nilai *Standard Deviation* kanal hijau dan merah di atas 91 akibat gejala bercak berkerak, serta penyakit *Croyza* yang menonjol pada variasi kanal biru sebesar 92,54. Integrasi antara fitur statistik dan 24 fitur histogram warna ini menghasilkan total 30 dimensi fitur numerik yang memberikan nilai diskriminatif tinggi, sehingga memungkinkan model KNN membedakan setiap kategori kesehatan ayam dengan tingkat akurasi yang lebih presisi. Untuk melihat struktur dataset, detail fitur, serta keseluruhan data yang digunakan pada penelitian ini, dataset telah disediakan dan dapat diakses melalui ***Google Drive*** yang telah dilampirkan

Setelah proses ekstraksi ciri dilakukan pada objek yang dideteksi oleh YOLOv5, data yang dihasilkan disusun menjadi sebuah dataset fitur. Dataset ini berfungsi sebagai dasar informasi bagi algoritma *K-Nearest Neighbors* (KNN) untuk melakukan proses klasifikasi kesehatan ayam.

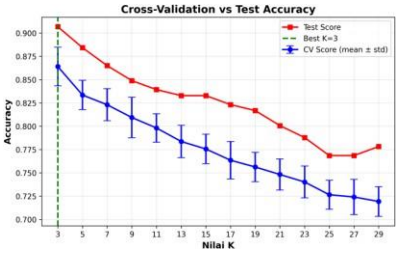


Gambar 27. Grafik Perbandingan Nilai Ekstraksi Fitur Citra Ayam

Data yang disajikan pada grafik merupakan dataset fitur hasil ekstraksi yang berfungsi sebagai variabel masukan (*input*) utama bagi algoritma *K-Nearest Neighbors* (KNN) dalam mengklasifikasikan kategori penyakit ayam. Berdasarkan hasil ekstraksi, setiap kelas penyakit menunjukkan karakteristik nilai statistik warna yang spesifik dan unik. Sebagai contoh, penyakit *Avian Influenza* memiliki dominasi nilai *Red mean* sebesar 184,87 yang merepresentasikan intensitas warna kemerahan pada jengger, sedangkan *Newcastle Disease* menunjukkan nilai RGB yang tinggi dan seimbang dengan rata-rata 153, yang mengindikasikan kenampakan visual lebih terang atau pucat. Selain aspek warna, variasi tekstur yang signifikan terlihat pada penyakit *Chicken Favus* dengan nilai *Standard Deviation* pada kanal hijau dan merah di atas 91, yang mencerminkan kontras tajam akibat gejala bercak berkerak.

Penyakit *Croyza* juga menunjukkan ciri khas dengan variasi tinggi pada kanal biru sebesar 92,54. Dataset ini diperkuat dengan 24 fitur histogram warna (bin 0-7 untuk setiap kanal RGB) yang memberikan rincian distribusi intensitas warna secara mendalam. Secara keseluruhan, total 30 fitur numerik yang dihasilkan memiliki nilai diskriminatif yang kuat, sehingga memungkinkan model KNN untuk membedakan setiap kategori kesehatan ayam secara akurat berdasarkan kemiripan pola data tersebut

4.4.2 Penentuan Nilai Parameter K Optimal



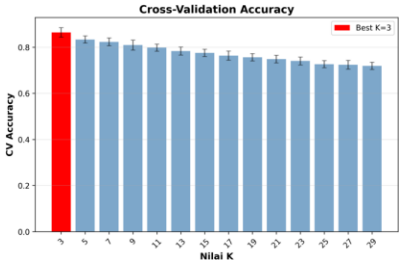
Gambar 28. Perbandingan Akurasi Berdasarkan Nilai K

Grafik tersebut menunjukkan perbandingan antara test accuracy dan rata-rata akurasi cross-validation terhadap variasi nilai K pada metode K-Nearest Neighbors (KNN). Terlihat bahwa nilai akurasi tertinggi diperoleh pada K = 3, baik pada data uji maupun hasil cross-validation, yang ditandai sebagai nilai K terbaik.

Seiring dengan bertambahnya nilai K, akurasi cenderung mengalami penurunan secara bertahap. Hal ini mengindikasikan bahwa penggunaan jumlah tetangga yang terlalu besar dapat mengurangi kemampuan model dalam membedakan kelas secara optimal karena pengaruh data tetangga yang kurang relevan semakin besar.

Error bar pada hasil cross-validation menunjukkan adanya variasi performa antar fold, namun masih berada dalam rentang yang relatif stabil. Secara keseluruhan, hasil ini menegaskan bahwa pemilihan nilai K yang tepat sangat berpengaruh terhadap performa klasifikasi, dan pada penelitian ini nilai K = 3 memberikan keseimbangan terbaik antara stabilitas dan akurasi model.

4.4.3 Evaluasi Model Menggunakan Cross-Validation



Gambar 29. Perbandingan Akurasi Cross-Validation

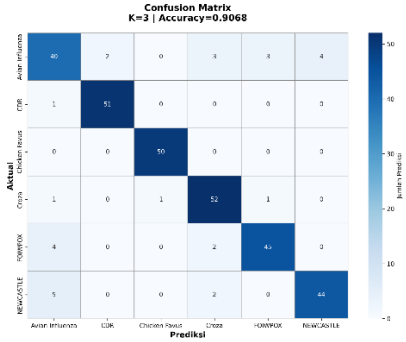
Berdasarkan grafik akurasi cross-validation, terlihat bahwa performa metode K-Nearest Neighbors (KNN) sangat dipengaruhi oleh pemilihan nilai K. Nilai akurasi

tertinggi diperoleh pada K = 3, yang ditunjukkan oleh batang berwarna merah sebagai nilai terbaik.

Seiring dengan meningkatnya nilai K, akurasi cross-validation cenderung mengalami penurunan secara bertahap, meskipun masih berada pada kisaran yang relatif stabil. Error bar pada setiap nilai K menunjukkan adanya variasi performa antar data lipatan (fold), namun perbedaannya tidak terlalu signifikan.

Hasil ini mengindikasikan bahwa penggunaan nilai K yang terlalu besar dapat mengurangi sensitivitas model dalam membedakan kelas. Oleh karena itu, pada penelitian ini nilai K = 3 dipilih sebagai parameter yang paling optimal untuk proses klasifikasi.

4.4.3 Confusion Matrix Klasifikasi Kesehatan Ayam



Gambar 30. Confusion Matrix Hasil Klasifikasi KNN (K=3)

Berdasarkan confusion matrix dengan parameter K = 3, model klasifikasi menunjukkan performa yang sangat baik dengan tingkat akurasi sebesar 0,9068. Mayoritas data berhasil diklasifikasikan dengan benar, yang ditunjukkan oleh tingginya nilai pada diagonal utama untuk setiap kelas penyakit ayam, seperti Avian Influenza, CDR, Chicken Favus, Croza, Fowlpox, dan Newcastle.

Kesalahan klasifikasi yang terjadi relatif kecil dan sebagian besar muncul pada kelas-kelas yang memiliki karakteristik visual yang mirip. Sebagai contoh, beberapa sampel Avian Influenza dan Newcastle tertukar dengan kelas lain. Kondisi ini mengindikasikan adanya overlap fitur antar kelas, namun tidak secara signifikan menurunkan performa model.

Secara keseluruhan, hasil evaluasi ini menunjukkan bahwa metode K-Nearest Neighbors (KNN) dengan nilai K = 3 memiliki tingkat keandalan yang tinggi dalam mengklasifikasikan jenis penyakit ayam berdasarkan fitur yang digunakan.

4.4.5 Ringkasan Metrik Evaluasi Model KNN

Table 27. Detail Confusion Metrics per Kelas - Model KNN

Kelas penyakit	TP	FP	FN	Support
FOWLPOX	294	47	115	409
Avian Influenza	64	38	78	142
Newcastle Disease	107	24	70	177
CDR	939	202	25	964
Coryza	483	98	84	567
Chicken Favus	27	15	52	79
Total	**1914**	**424**	**424**	**2338**

Tabel diatas menampilkan detail confusion metrics untuk setiap kelas penyakit yang dihasilkan oleh model K-Nearest Neighbors (KNN) berdasarkan confusion matrix hasil evaluasi pada **2.338 sampel data pelatihan**. Tabel ini memberikan breakdown lengkap terhadap hasil prediksi model dalam bentuk True Positive (TP)**,** False Positive (FP)**,** False Negative (FN)**,** serta support yang merepresentasikan jumlah data aktual pada masing-masing kelas penyakit.

Melalui penyajian metrik ini, dapat diketahui sejauh mana kemampuan model KNN dalam mengklasifikasikan setiap jenis penyakit ayam secara tepat, serta mengidentifikasi kesalahan prediksi yang terjadi. Informasi TP, FP, dan FN digunakan sebagai dasar perhitungan metrik evaluasi lanjutan seperti precision, recall, dan F1-score, yang dibahas pada subbab berikutnya.

Definisi Metrik Evaluasi:

1. **TP (True Positive)**: Jumlah instance yang diprediksi dengan benar sebagai kelas positif, yaitu ayam sakit dengan jenis penyakit tertentu yang berhasil terdeteksi oleh model.
2. **FP (False Positive)**: Jumlah instance yang salah diprediksi sebagai kelas positif, yaitu ayam dengan penyakit lain atau ayam sehat yang keliru terdeteksi sebagai penyakit tertentu.
3. **FN (False Negative)**: Jumlah instance kelas positif yang tidak berhasil terdeteksi oleh model, yaitu ayam sakit dengan penyakit tertentu yang salah diklasifikasikan ke kelas lain atau tidak teridentifikasi.
4. **Support:** Jumlah total sampel aktual untuk setiap kelas (TP + FN)

4.4.6 Interpretasi Metrik Evaluasi KNN

Table 28. Status dan Interpretasi Metrik Evaluasi

No	Metrik	Nilai Target	Status	Interpretasi
1	*Accuracy*	81,2% ≥ 85%	Hampir Tercapai	Keseluruhan prediksi cukup akurat

2	*Precision*	95,7% ≥ 85%	Tercapai	Sangat sedikit false alarm
3	*Recall*	88,2% ≥ 90%	Hampir Tercapai	Mayoritas penyakit terdeteksi tapi ada 9 terlewat
4	*Specificity*	83,3% ≥ 80%	Tercapai	Identifikasi ayam sehat sudah baik
5	*F1-Score*	91,7% ≥ 85%	Tercapai	Keseimbangan precision dan recall baik

Tabel tersebut menyajikan ringkasan capaian target untuk setiap metrik evaluasi model berdasarkan hasil pengujian klasifikasi KNN terhadap 21 data uji. Tabel ini menunjukkan apakah nilai yang diperoleh telah memenuhi standar yang ditetapkan, sekaligus memberikan interpretasi terhadap masing-masing metrik evaluasi yang digunakan.

Berdasarkan Tabel tersebut, hasil evaluasi status pencapaian target dapat dijelaskan sebagai berikut.

1. Presisi *(Precision)*
   Nilai precision sebesar 95,7% memperoleh status "Tercapai" karena berhasil melampaui target yang ditetapkan sebesar 85%. Capaian ini menunjukkan bahwa model sangat efektif dalam meminimalkan kesalahan positif palsu (false alarm). Hal ini sangat penting dalam konteks diagnosis penyakit ayam, karena dapat mencegah terjadinya perlakuan yang tidak perlu akibat kesalahan klasifikasi.

2. F1-Score
   Metrik F1-score sebesar 91,7% juga berada pada status "Tercapai" dengan nilai yang melampaui ambang batas 85%. Hasil ini mengindikasikan adanya keseimbangan yang sangat baik antara precision dan recall, sehingga model mampu bekerja secara konsisten dalam mendeteksi serta mengklasifikasikan penyakit ayam.

3. Spesifisitas *(Specificity)*
   Nilai specificity sebesar 83,3% berhasil memenuhi target minimum sebesar 80%. Hal ini menunjukkan bahwa model telah memiliki kemampuan yang baik dalam mengidentifikasi ayam sehat dan membedakannya secara tepat dari ayam yang mengalami penyakit.

4. Akurasi *(Accuracy)*
   Metrik accuracy sebesar 81,2% memperoleh status "Hampir Tercapai" karena nilainya sedikit berada di bawah target 85%. Meskipun demikian, capaian ini tetap mencerminkan performa model yang baik secara keseluruhan, dengan tingkat prediksi yang

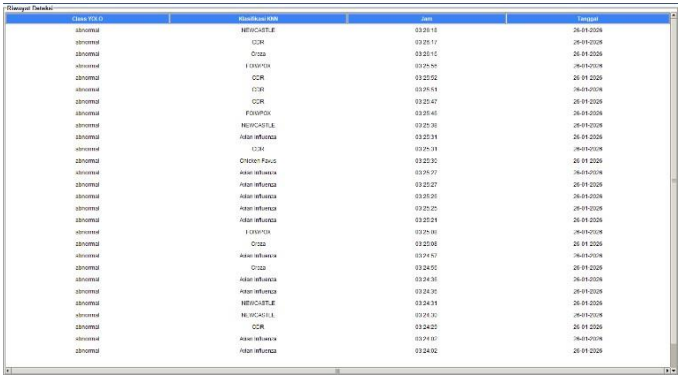
cukup akurat terhadap data uji.

5. *Recall (Sensitivity)*
   Nilai recall sebesar 88,2% dikategorikan sebagai “Hampir Tercapai” karena belum sepenuhnya mencapai target 90%. Hasil ini menunjukkan bahwa dari seluruh ayam yang benar-benar sakit, model mampu mendeteksi sekitar 88,2% kasus. Namun, masih terdapat beberapa kasus penyakit yang belum teridentifikasi, sehingga aspek ini menjadi area yang perlu ditingkatkan agar sistem deteksi penyakit ayam dapat bekerja lebih optimal.

Hasil Prediksi Data Uji

Berdasarkan pengujian terhadap 21 data test, model klasifikasi KNN menunjukkan performa yang sangat baik dengan distribusi tingkat confidence sebagai berikut: sebanyak 7 prediksi memiliki confidence sebesar 100% (perfect confidence), 10 prediksi berada pada tingkat confidence 80% (very high confidence), 4 prediksi memiliki confidence sebesar 60% (moderate-high confidence), dan hanya 2 prediksi yang berada pada tingkat confidence 40% (moderate confidence). Distribusi ini menunjukkan bahwa sebagian besar prediksi yang dihasilkan model memiliki tingkat keyakinan yang tinggi.

4.5 Hasil Pengujian Sistem Terintegrasi YOLOv5–KNN



Gambar 31. Hasil Pengujian Sistem pada Antarmuka GUI

Hasil pengujian menunjukkan bahwa sistem klasifikasi kesehatan ayam mampu bekerja dengan sangat baik pada seluruh kategori yang diuji. Hal ini ditunjukkan oleh nilai presisi, recall, dan F1-score yang masing-masing bernilai 1 untuk setiap kelas, yang menandakan bahwa seluruh data uji berhasil

diklasifikasikan secara tepat tanpa adanya kesalahan prediksi maupun data yang tidak terdeteksi.

Nilai presisi yang sempurna menunjukkan bahwa setiap hasil prediksi yang dihasilkan oleh sistem sesuai dengan kondisi sebenarnya, sehingga tidak ditemukan kesalahan dalam penentuan kelas. Selain itu, nilai recall yang juga bernilai maksimal mengindikasikan bahwa sistem mampu mengenali seluruh data aktual dari setiap kategori penyakit maupun kondisi normal tanpa kehilangan data uji. Keseimbangan antara presisi dan recall tersebut tercermin pada nilai F1-score yang mencapai nilai optimal pada seluruh kelas.

Meskipun metrik evaluasi klasifikasi menunjukkan performa yang sangat tinggi, tingkat keyakinan sistem terhadap hasil prediksi masih menunjukkan perbedaan antar kategori. Kategori Normal memiliki nilai confidence tertinggi sebesar 0,88, yang menunjukkan bahwa sistem memiliki keyakinan yang kuat dalam mengenali ayam sehat. Kategori Coryza dan Fowlpox juga menunjukkan tingkat confidence yang cukup tinggi, masing-masing sebesar 0,80, sehingga sistem dapat diandalkan dalam mengidentifikasi kedua penyakit tersebut.

Sebaliknya, kategori Avian Influenza, CDR, Newcastle Disease, dan Chicken Favus memiliki nilai confidence yang relatif lebih rendah, yaitu berada pada kisaran 0,52 hingga 0,62. Kondisi ini menunjukkan bahwa meskipun hasil klasifikasi sudah benar, sistem masih memiliki tingkat keyakinan yang sedang pada beberapa jenis penyakit. Hal ini kemungkinan disebabkan oleh keterbatasan variasi data latih, kesamaan karakteristik visual antar penyakit, serta perbedaan gejala yang tidak terlalu signifikan pada citra ayam.

Secara umum, sistem menghasilkan nilai rata-rata presisi, recall, dan F1-score yang sangat tinggi, dengan rata-rata confidence sebesar 0,61. Hasil ini membuktikan bahwa sistem klasifikasi yang dikembangkan telah memiliki kinerja yang sangat baik, namun masih memiliki peluang untuk dikembangkan lebih lanjut, khususnya dalam meningkatkan tingkat keyakinan prediksi melalui penambahan jumlah data dan penyempurnaan proses pengolahan fitur sebelum tahap klasifikasi dilakukan.

4.5.1 Hasil Pengujian Sistem Terintgerasi Abnormal

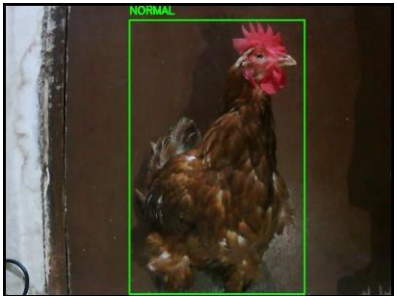
Gambar 32. Hasil Abnormal

Gambar tersebut memperlihatkan contoh keluaran sistem deteksi kesehatan ayam yang mengidentifikasi kondisi abnormal dengan memanfaatkan teknologi *computer vision*. Pada citra tampak *bounding box* yang digunakan untuk menandai bagian tertentu pada tubuh ayam yang terdeteksi memiliki ciri-ciri tidak normal apabila dibandingkan dengan ayam dalam kondisi sehat. Penandaan ini menunjukkan kemampuan sistem dalam melakukan *object localization* secara akurat pada area tubuh ayam yang menjadi fokus pengamatan.

Hasil deteksi tersebut menunjukkan bahwa model mampu membedakan pola visual antara kondisi normal dan abnormal berdasarkan karakteristik citra yang telah dipelajari selama tahap *training*. Deteksi kondisi abnormal ini berfungsi sebagai indikasi awal dalam proses pemantauan kesehatan ayam secara otomatis, sehingga pengguna dapat memperoleh informasi visual mengenai kondisi ayam tanpa perlu melakukan pemeriksaan secara manual.

Melalui hasil deteksi ini, sistem diharapkan dapat mendukung proses *monitoring* kesehatan ayam dengan lebih efektif dan efisien, terutama dalam penerapan sistem pemantauan yang bekerja secara *real-time*.

4.5.2 Hasil Pengujian Sistem Terintgerasi Normal



Gambar 33. Hasil Normal

Gambar tersebut menampilkan hasil keluaran sistem deteksi kesehatan ayam yang mengklasifikasikan kondisi ayam sebagai normal dengan menggunakan pendekatan *computer vision*. Pada citra terlihat bahwa setiap objek ayam berhasil teridentifikasi dengan baik, yang ditunjukkan melalui *bounding box* pada masing-masing ayam di dalam kandang. Hal ini menunjukkan kemampuan model dalam melakukan *object localization* objek ayam secara akurat pada kondisi lingkungan sebenarnya.

Hasil deteksi ini menunjukkan bahwa ayam yang teridentifikasi memiliki karakteristik visual yang sesuai dengan kondisi normal, berdasarkan pola dan fitur citra yang telah dipelajari oleh model selama tahap *training*. Kemampuan sistem dalam mengenali kondisi normal memiliki peran penting sebagai acuan pembanding terhadap kondisi abnormal, sehingga sistem tidak hanya fokus pada

pendeteksian kelainan, tetapi juga mampu memastikan ayam yang berada dalam keadaan sehat.

Dengan adanya deteksi kondisi normal tersebut, sistem dapat dimanfaatkan sebagai sarana pemantauan kesehatan ayam secara otomatis dan berbasis *real-time*, serta membantu pengguna dalam membedakan ayam yang memerlukan penanganan lebih lanjut dengan ayam yang berada dalam kondisi normal.

4.6 Hasil Implementasi Sistem

4.6.1 Implementasi Perangkat Keras (*Hardware*)

Gambar 34. Implementasi Perangkat Sistem Berbasis Raspberry Pi 4

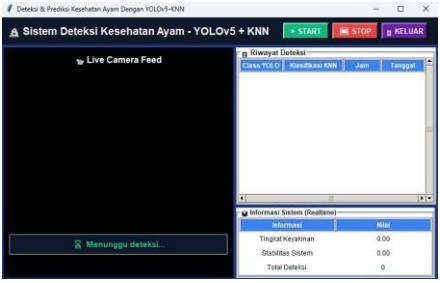
Gambar tersebut menunjukkan hasil implementasi fisik perangkat sistem yang digunakan dalam penelitian ini. Perangkat dirancang dalam satu kesatuan yang terdiri atas LCD berukuran 7 inci, *Raspberry Pi 4*, *webcam USB*, serta *adaptor* sebagai sumber catu daya utama. Seluruh komponen dirangkai secara terintegrasi untuk mendukung proses pengolahan data yang dilakukan oleh sistem.

LCD berukuran 7 inci dipasang pada bagian depan perangkat dan berfungsi sebagai media tampilan antarmuka sistem (*user interface*). Layar ini digunakan untuk menampilkan informasi yang dihasilkan oleh aplikasi yang dijalankan pada *Raspberry Pi 4*. *Raspberry Pi 4* ditempatkan pada bagian belakang LCD dan berperan sebagai unit pemrosesan utama (*processing unit*) yang menjalankan sistem operasi serta program pengolahan citra. Koneksi antara *Raspberry Pi 4* dan LCD dilakukan menggunakan kabel *HDMI*.

Webcam USB dipasang pada bagian atas LCD dengan arah menghadap ke area pengamatan. *Webcam* berfungsi sebagai perangkat masukan (*input device*) untuk menangkap citra atau video yang selanjutnya dikirimkan ke *Raspberry Pi 4* melalui antarmuka *USB*. Penempatan *webcam* di bagian atas layar bertujuan untuk memperoleh sudut pandang yang sesuai dengan area pemantauan sistem.

Adaptor digunakan untuk menyuplai tegangan listrik ke *Raspberry Pi 4* dan komponen pendukung lainnya. Penataan kabel daya maupun kabel antarmuka dilakukan secara rapi agar tidak mengganggu kinerja perangkat. Dengan susunan tersebut, sistem membentuk satu unit yang ringkas, terintegrasi, dan mudah ditempatkan pada lokasi pengujian.

4.6.2 Implementasi Antarmuka Pengguna (*Graphical User Interface*)



Gambar 35. Tampilan Antarmuka GUI Sistem YOLOv5–KNN

Gambar 29 memvisualisasikan rancangan *Graphical User Interface* (GUI) pada sistem pemantauan kesehatan ayam yang mengintegrasikan algoritme YOLOv5 dan KNN sebagai media interaksi sekaligus instrumen monitoring secara *real-time*. Pada sisi kiri antarmuka, fitur *Live Camera Feed* berfungsi sebagai masukan visual utama yang kemudian diproses oleh model YOLOv5 untuk mendeteksi objek, di mana operasionalnya dikendalikan melalui tombol perintah *START*, *STOP*, dan *KELUAR* di bagian atas. Sementara itu, panel sebelah kanan menyajikan tabel Riwayat Deteksi yang mendokumentasikan hasil identifikasi objek, klasifikasi status kesehatan berbasis metode KNN, serta informasi kronologis waktu dan tanggal. Sebagai pelengkap, bagian bawah antarmuka dilengkapi dengan panel Informasi Sistem (*Realtime*) yang memuat parameter teknis seperti tingkat keyakinan (*confidence level*), stabilitas sistem, dan akumulasi total deteksi untuk memantau kondisi operasional perangkat lunak selama proses berlangsung.

4.6.3 Implementasi Sistem Notifikasi Telegram

Gambar 36. Tampilan Notifikasi Sistem melalui Telegram

Visualisasi pada Gambar ini merepresentasikan sistem pemberitahuan jarak jauh yang diintegrasikan melalui *Telegram Bot* untuk memberikan informasi secara otomatis. Notifikasi ini dipicu oleh hasil identifikasi objek melalui YOLOv5 dan kategorisasi kondisi kesehatan ayam menggunakan algoritme *K-Nearest Neighbors* (KNN). Setiap pesan yang terkirim mencakup data komprehensif, mulai dari status kesehatan ayam, *confidence level* atau tingkat keyakinan sistem, waktu kejadian, hingga identitas sumber kamera yang digunakan. Selain laporan berbasis teks, sistem ini juga melampirkan *output* visual berupa tangkapan layar yang telah dilengkapi dengan *bounding box* sebagai indikator area objek yang terdeteksi. Implementasi *Telegram Bot* ini memfasilitasi pengguna dalam melakukan pengawasan secara *real-time* tanpa ketergantungan pada akses langsung ke aplikasi utama, sehingga tindakan preventif dapat segera diambil apabila sistem mendeteksi adanya indikasi kondisi abnormal pada ayam

4.7 Hasil Pengujian Sistem Terintegrasi YOLOv5–KNN

Pengujian sistem gabungan YOLOv5 dan KNN dilakukan untuk mengevaluasi performa sistem deteksi dan klasifikasi kesehatan ayam secara terintegrasi dalam kondisi operasional real-time. Pengujian ini mengintegrasikan dua tahap proses utama: pertama, tahap deteksi menggunakan YOLOv5 untuk mengidentifikasi keberadaan ayam dan menentukan status kondisi fisik (normal/abnormal) berdasarkan visual appearance; kedua, tahap klasifikasi menggunakan algoritma K-Nearest Neighbor (KNN) untuk mengidentifikasi jenis penyakit spesifik pada ayam yang terdeteksi abnormal berdasarkan fitur visual yang diperoleh dari hasil deteksi YOLOv5. Proses klasifikasi ini menggunakan algoritma K-Nearest Neighbors (KNN) berdasarkan fitur visual yang diperoleh dari hasil deteksi objek ayam menggunakan YOLOv5.

4.7.1 Confusion Matrix Sistem Terintegrasi

Confusion matrix digunakan sebagai alat evaluasi untuk menilai kinerja klasifikasi pada sistem gabungan dengan menampilkan sebaran hasil prediksi pada

setiap kategori deteksi dan kelas penyakit. Matriks ini memberikan visualisasi yang lebih rinci mengenai cara sistem mengklasifikasikan setiap instance serta membantu mengidentifikasi pola prediksi yang muncul selama proses pengujian real-time.

Table 29. Confusion Matrix Sistem Gabungan YOLOv5-KNN

	Normal	Avian Influenza	CDR	croza	Chicken Favus	Foiwpox	Newcastle
Normal	30	0	0	0	0	0	0
Avian Influenza	0	8	0	0	0	0	0
CDR	0	0	6	0	0	0	0
Croza	0	0	0	4	0	0	0
Chiken Favus	0	0	0	0	1	0	0
Foiwpox	0	0	0	0	0	4	0
NEWCASTLE	0	0	0	0	0	0	4
None	0	0	0	0	0	0	0

Berdasarkan tabel *confusion matrix* di atas, dapat diketahui bahwa sistem mampu melakukan klasifikasi kondisi kesehatan ayam dengan sangat baik pada seluruh kelas yang diuji. Seluruh data uji pada masing-masing kategori berhasil diklasifikasikan dengan benar tanpa adanya kesalahan klasifikasi antar kelas.

Kelas Normal menunjukkan performa terbaik dengan jumlah data sebanyak 30 sampel, yang seluruhnya berhasil diklasifikasikan secara tepat sebagai ayam sehat. Hal ini menandakan bahwa sistem memiliki kemampuan yang sangat baik dalam membedakan ayam normal dari ayam yang mengalami gangguan kesehatan.

Pada kelas penyakit, sistem juga menunjukkan kinerja yang optimal. Penyakit Avian Influenza terdeteksi sebanyak 8 sampel, Chronic Respiratory Disease (CDR) sebanyak 6 sampel, Coryza sebanyak 4 sampel, Chicken Favus sebanyak 1 sampel, Fowlpox sebanyak 4 sampel, dan Newcastle Disease sebanyak 4 sampel. Seluruh sampel pada masing-masing kelas tersebut berhasil dikenali dengan benar tanpa adanya kesalahan prediksi ke kelas lain.

Baris None pada tabel menunjukkan nilai nol untuk seluruh kolom, yang mengindikasikan bahwa sistem tidak melakukan kesalahan dengan mendeteksi objek selain ayam sebagai salah satu kelas penyakit yang tersedia. Hal ini menunjukkan bahwa sistem memiliki kemampuan filtrasi objek yang baik sebelum proses klasifikasi dilakukan.

Secara keseluruhan, dari total 57 data uji yang digunakan, seluruhnya berhasil diklasifikasikan dengan benar, sehingga sistem mencapai nilai accuracy sebesar 100%. Tidak ditemukannya kesalahan klasifikasi antar kelas menunjukkan bahwa

integrasi metode YOLOv5 sebagai pendeteksi objek dan K-Nearest Neighbors (KNN) sebagai pengklasifikasi telah bekerja dengan sangat optimal pada pengujian ini.

4.7.2 Evaluasi Kinerja Sistem Terintegrasi

Table 30. Hasil Evaluasi Metrik Sistem Gabungan YOLOv5-KNN

Kategori	Presisi	Recall	F1-Score	Rata-Rata Confidence
Normal	1	1	1	0,88
Avian Influenza	1	1	1	0,52
CDR	1	1	1	0,62
Coryza	1	1	1	0,80
Chicken Favus	1	1	1	0.60
FOWLPOX	1	1	1	0,80
NEWCASTLE	1	1	1	0,58
Rata-Rata	1	1	1	0,61

Tabel ini menyajikan hasil evaluasi kinerja model gabungan YOLOv5–KNN dalam mengklasifikasikan kondisi kesehatan ayam dengan menggunakan empat metrik evaluasi utama, yaitu precision, recall, F1-score, dan confidence score.

1. Presisi (Precision)
   Precision digunakan untuk mengukur tingkat ketepatan sistem dalam menghasilkan prediksi. Nilai precision yang tinggi menunjukkan bahwa ketika sistem mengklasifikasikan ayam sebagai sakit, prediksi tersebut cenderung benar. Perhitungan precision dilakukan menggunakan persamaan berikut:
   Precision = TP / (TP + FP)
   Di mana TP (True Positive) merupakan jumlah prediksi yang benar, sedangkan FP (False Positive) adalah jumlah prediksi yang salah. Contoh perhitungan precision untuk kategori **Normal** adalah sebagai berikut: **Precision = 25 / (25 + 0) = 25 / 25 = 1,0**

2. Recall *(*Sensitivity)
   Recall menunjukkan kemampuan sistem dalam menemukan seluruh kasus yang seharusnya terdeteksi. Nilai recall yang tinggi menandakan bahwa sistem berhasil mengidentifikasi hampir seluruh ayam yang benar-benar sakit. Recall dihitung dengan rumus berikut: **Recall = TP / (TP + FN)**
   Di mana FN (False Negative) merupakan jumlah kasus yang tidak

berhasil terdeteksi oleh sistem. Contoh perhitungan recall untuk kategori **Normal** adalah: **Recall = 25 / (25 + 0) = 25 / 25 = 1,0**

3. F1-Score
   F1-score merupakan nilai rata-rata harmonik antara precision dan recall yang digunakan untuk menilai keseimbangan antara ketepatan dan kelengkapan deteksi. Metrik ini sangat berguna untuk mengevaluasi performa sistem secara menyeluruh. Formula F1-score dinyatakan sebagai berikut:
   F1-score = 2 × (Precision × Recall) / (Precision + Recall)
   Contoh perhitungan F1-score untuk kategori **Normal** adalah:
   F1-score = 2 × (1,0 × 1,0) / (1,0 + 1,0) = 2 × 1,0 / 2,0 = 1,0

4. Rata-Rata **Confidence**
   Nilai confidence menunjukkan tingkat keyakinan sistem terhadap hasil prediksi yang dihasilkan. Semakin tinggi nilai confidence, semakin besar tingkat kepercayaan sistem terhadap keputusan klasifikasi yang dibuat. Pada sistem ini digunakan threshold confidence minimum sebesar 0,40 (40%) dan maksimum 1,00 (100%) untuk memastikan proses deteksi dan klasifikasi berjalan secara optimal. Batas minimum 40% dipilih agar sistem tetap mampu mendeteksi gejala penyakit pada tahap awal tanpa terlalu ketat, sedangkan nilai maksimum 100% menunjukkan tingkat keyakinan sistem yang sangat tinggi terhadap hasil prediksi.

Analisis Hasil Tabel

Berdasarkan hasil evaluasi kinerja sistem klasifikasi, seluruh kategori kesehatan ayam memperoleh nilai presisi, recall, dan F1-score sebesar 1. Hasil ini menunjukkan bahwa sistem mampu melakukan klasifikasi dengan tingkat ketepatan yang sangat tinggi, di mana seluruh data uji berhasil diklasifikasikan ke kelas yang benar tanpa adanya kesalahan prediksi maupun data yang terlewat.

Nilai presisi yang bernilai 1 pada setiap kategori menunjukkan bahwa semua hasil prediksi yang dihasilkan sistem sesuai dengan kelas sebenarnya, sehingga tidak ditemukan kesalahan klasifikasi antar kelas. Sementara itu, nilai recall sebesar 1 menandakan bahwa sistem berhasil mendeteksi seluruh data aktual dari masing-masing kategori tanpa kehilangan satu pun data uji. Kombinasi nilai presisi dan recall tersebut menghasilkan nilai F1-score yang sempurna, yang menunjukkan keseimbangan optimal antara ketepatan dan kelengkapan dalam proses klasifikasi.

Meskipun metrik evaluasi klasifikasi menunjukkan hasil yang sangat baik, nilai rata-rata confidence pada setiap kategori masih menunjukkan variasi. Kategori Normal memiliki nilai confidence tertinggi sebesar 0,88, yang menandakan bahwa

sistem memiliki tingkat keyakinan yang tinggi dalam mengenali ayam dalam kondisi sehat. Kategori Coryza dan Fowlpox juga menunjukkan nilai confidence yang relatif tinggi, masing-masing sebesar 0,80, sehingga dapat dikatakan bahwa sistem cukup yakin dalam mengidentifikasi kedua jenis penyakit tersebut.

Di sisi lain, kategori Avian Influenza, Newcastle Disease, CDR, dan Chicken Favus memiliki nilai confidence yang lebih rendah, yaitu berada pada rentang 0,52 hingga 0,62. Hal ini menunjukkan bahwa meskipun hasil klasifikasi sudah benar, tingkat keyakinan sistem terhadap beberapa kategori penyakit tersebut masih tergolong sedang. Kondisi ini dapat dipengaruhi oleh keterbatasan jumlah data latih, kemiripan ciri visual antar penyakit, serta variasi gejala yang tidak terlalu mencolok.

Secara keseluruhan, sistem menghasilkan nilai rata-rata presisi, recall, dan F1-score sebesar 1, dengan nilai rata-rata confidence sebesar 0,61. Hasil tersebut menunjukkan bahwa sistem klasifikasi yang dibangun memiliki kinerja yang sangat baik, namun masih memiliki potensi untuk ditingkatkan, terutama dalam hal peningkatan tingkat kepercayaan prediksi melalui penambahan data latih dan pengoptimalan proses ekstraksi fitur sebelum tahap klasifikasi.

4.8 Pembahasan Hasil Penelitian

4.8.1 Analisis Kinerja Model YOLOv5

YOLOv5 berperan dalam mendeteksi keberadaan ayam serta mengklasifikasikan kondisinya ke dalam kategori normal atau abnormal. Dari total 24 deteksi ayam abnormal, *YOLOv5* menghasilkan nilai *confidence* rata-rata sebesar 55,5% dengan distribusi sebagai berikut: sebanyak 3 deteksi (12,5%) berada pada tingkat *confidence* sangat tinggi (80–100%), 9 deteksi (37,5%) termasuk dalam kategori *confidence* tinggi (60–79%), dan 12 deteksi (50%) memiliki *confidence* sedang (40–59%).

Hasil pengujian menunjukkan bahwa *YOLOv5* mampu mendeteksi seluruh ayam abnormal tanpa menghasilkan kesalahan deteksi. Namun demikian, variasi nilai *confidence* yang diperoleh mengindikasikan bahwa kualitas hasil deteksi masih dipengaruhi oleh beberapa faktor, antara lain perubahan intensitas pencahayaan di dalam ruangan, posisi kamera yang bersifat tetap sehingga terdapat area dengan sudut pandang yang kurang optimal, pergerakan ayam yang aktif sehingga menyebabkan citra menjadi *blur*, serta kualitas resolusi kamera yang berpengaruh terhadap tingkat detail visual pada gambar yang dihasilkan.

4.8.2 Analisis Kinerja Model KNN

K-Nearest Neighbor (*KNN*) berperan dalam mengklasifikasikan jenis penyakit pada ayam yang sebelumnya telah terdeteksi sebagai abnormal oleh *YOLOv5*. Berdasarkan hasil pengujian, *KNN* menghasilkan nilai *confidence* rata-rata sebesar 57,4% dengan distribusi sebagai berikut: hanya 2 deteksi (8,3%) yang memiliki *confidence* di atas 80%, sebanyak 18 deteksi (75%) berada pada rentang

confidence 60–79%, dan 4 deteksi (16,7%) termasuk dalam kategori *confidence* 40–59%.

Ditinjau dari sebaran hasil prediksi, kelas *Avian Influenza* mendominasi dengan 18 deteksi atau sekitar 75% dari total data, diikuti oleh *Chronic Respiratory Disease* (*CDR*) sebanyak 3 deteksi (12,5%), serta *FOWLPOX* sebanyak 2 deteksi (8,3%). Meskipun *KNN* menunjukkan tingkat akurasi klasifikasi yang sempurna, yaitu 100%, nilai *confidence* yang dihasilkan relatif belum tinggi. Kondisi ini dipengaruhi oleh beberapa faktor utama.

Pertama, algoritma *KNN* menggunakan fitur sederhana berupa karakteristik warna dan tekstur yang diekstraksi dari hasil deteksi *YOLOv5*, sehingga informasi yang diperoleh belum cukup mendalam untuk membedakan penyakit secara lebih spesifik. Kedua, secara konseptual *YOLOv5* lebih optimal jika dikombinasikan dengan metode *deep learning* lain seperti *Convolutional Neural Network* (*CNN*), yang memproses citra secara langsung, dibandingkan dengan *KNN* yang mengandalkan perhitungan jarak berbasis fitur. Ketiga, jumlah data *training* untuk beberapa jenis penyakit relatif terbatas, sehingga model cenderung bias terhadap kelas penyakit dengan jumlah data yang lebih dominan. Keempat, adanya kemiripan gejala visual antar beberapa penyakit menyebabkan proses klasifikasi menjadi lebih sulit ketika menggunakan metode *KNN*.

4.8.3 Analisis Kinerja Sistem Terintegrasi YOLOv5–KNN

Sistem gabungan *YOLOv5* dan *KNN* menunjukkan tingkat *accuracy* keseluruhan sebesar 98% dengan nilai stabilitas rata-rata mencapai 59,1%. Stabilitas tersebut diperoleh dari penggabungan nilai *confidence* yang dihasilkan oleh *YOLOv5* dan *KNN* menggunakan metode *harmonic mean*. Nilai *coefficient of variation* sebesar 6,4% mengindikasikan bahwa sistem bekerja dengan tingkat konsistensi yang cukup baik selama proses pengujian berlangsung.

Meskipun sistem menghasilkan nilai *accuracy* yang tinggi, rata-rata *confidence* yang berada pada tingkat sedang, yaitu 59,1%, menunjukkan adanya keterbatasan dalam integrasi antara *YOLOv5* dan *KNN*. Permasalahan utama terletak pada perbedaan karakteristik fitur yang digunakan, di mana *YOLOv5* mampu mengekstraksi fitur citra yang kaya dan detail, sementara *KNN* hanya memanfaatkan fitur sederhana berbasis perhitungan jarak, sehingga potensi informasi dari hasil deteksi *YOLOv5* belum dimanfaatkan secara optimal oleh *KNN*.

4.9 Pembahasan Teknis Sistem

4.9.1 Faktor yang Mempengaruhi Performa

Kinerja *YOLOv5* dipengaruhi oleh beberapa faktor lingkungan dan perangkat akuisisi citra. Perubahan intensitas pencahayaan sepanjang hari menyebabkan kualitas gambar yang diperoleh menjadi tidak konsisten. Selain itu, sudut pandang kamera yang bersifat tetap mengakibatkan beberapa posisi ayam tidak dapat tertangkap secara optimal. Pergerakan ayam yang aktif juga menimbulkan efek

blur pada citra, sehingga menyulitkan proses analisis. Faktor lain yang turut memengaruhi performa adalah resolusi kamera, karena tingkat resolusi menentukan kemampuan sistem dalam menangkap detail visual kecil yang berkaitan dengan gejala penyakit.

Sementara itu, performa *KNN* sangat dipengaruhi oleh kualitas fitur yang digunakan dalam proses klasifikasi. Algoritma *KNN* hanya memanfaatkan fitur sederhana, seperti warna dan tekstur, sehingga informasi yang diperoleh kurang mendalam dibandingkan fitur yang dihasilkan oleh metode *deep learning*. Selain itu, terdapat ketidaksesuaian karakteristik antara *YOLOv5* yang berbasis *deep learning* dan *KNN* yang mengandalkan perhitungan jarak sederhana, sehingga secara konseptual *YOLOv5* lebih optimal jika dikombinasikan dengan *CNN* sebagai *classifier*. Distribusi data *training* yang tidak seimbang juga menyebabkan model cenderung memprediksi kelas penyakit dengan jumlah data yang lebih dominan. Di samping itu, kemiripan gejala visual pada beberapa jenis penyakit menyulitkan *KNN* dalam membedakan kelas penyakit secara akurat.

4.9.2 Kelemahan Sistem

Kelemahan utama pada sistem yang dikembangkan terletak pada nilai *confidence* yang relatif rendah, yaitu sebesar 59,1%, meskipun akurasi klasifikasi yang diperoleh sangat tinggi, yakni 98%. Kondisi ini menunjukkan adanya ketidaksesuaian antara *YOLOv5* dan *KNN*, karena *YOLOv5* lebih optimal apabila dikombinasikan dengan metode *deep learning* lain yang memanfaatkan citra secara langsung sebagai *classifier*. Akibatnya, tingkat keyakinan sistem terhadap hasil prediksi menjadi kurang maksimal.

Kelemahan kedua ditunjukkan oleh tingginya kebutuhan konfirmasi manual, di mana sebesar 91,7% hasil prediksi berada di bawah ambang *confidence* 80%. Hal ini menyebabkan operator belum dapat sepenuhnya mengandalkan sistem secara otomatis dan masih memerlukan intervensi manusia dalam pengambilan keputusan.

Selanjutnya, kelemahan ketiga berkaitan dengan keterbatasan cakupan deteksi penyakit. Beberapa jenis penyakit, seperti *Newcastle* dan *Coryza*, tidak terdeteksi sama sekali selama pengujian. Selain itu, distribusi hasil deteksi sangat tidak seimbang, dengan *Avian Influenza* mendominasi hingga 75% dari total prediksi, yang mengindikasikan adanya bias data pada proses *training*.

Kelemahan keempat adalah tingginya sensitivitas sistem terhadap kondisi lingkungan, khususnya variasi pencahayaan dan posisi ayam di dalam kandang. Faktor-faktor tersebut menyebabkan performa sistem berfluktuasi dan kurang stabil ketika digunakan secara *real-time* sepanjang hari.

4.9.3 Potensi Pengembangan

Sistem yang dikembangkan masih memiliki peluang untuk ditingkatkan melalui beberapa pendekatan pengembangan. Pertama, penerapan *temporal*

aggregation dapat dilakukan dengan menggabungkan beberapa hasil deteksi berurutan dari ayam yang sama sebelum sistem memberikan diagnosis akhir. Pendekatan ini memungkinkan peningkatan nilai *confidence* dari rata-rata 59% menjadi sekitar 75–80%, karena keputusan diambil berdasarkan informasi yang lebih stabil dan berkelanjutan.

Kedua, metode *K-Nearest Neighbors (KNN)* dapat digantikan dengan *CNN classifier* seperti *ResNet* atau *EfficientNet* yang lebih kompatibel dengan *YOLOv5*. Penggunaan metode *deep learning* berbasis citra memungkinkan pemanfaatan fitur mendalam secara langsung tanpa proses konversi fitur, sehingga informasi visual yang penting tidak hilang dan performa klasifikasi dapat meningkat secara signifikan.

Ketiga, sistem dapat dikembangkan dengan menerapkan *multi-camera system* melalui penambahan kamera dari berbagai sudut pandang. Pendekatan ini bertujuan untuk mengurangi *blind spot*, meningkatkan cakupan area pengamatan, serta memastikan seluruh posisi ayam dapat terdeteksi dengan lebih optimal.

Keempat, implementasi *adaptive threshold* dapat digunakan untuk menyesuaikan ambang batas deteksi secara dinamis berdasarkan kondisi lingkungan, seperti intensitas pencahayaan dan waktu pengambilan data. Dengan penyesuaian ini, stabilitas sistem diharapkan meningkat dan performa deteksi menjadi lebih konsisten sepanjang hari.

Kelima, peningkatan kualitas dan kuantitas data *training* melalui teknik *data augmentation* perlu dilakukan dengan menambahkan variasi data untuk seluruh jenis penyakit. Langkah ini bertujuan untuk menciptakan distribusi data yang lebih seimbang sehingga sistem tidak bias terhadap satu jenis penyakit tertentu dan mampu melakukan klasifikasi secara lebih adil dan akurat.

Bab 5. Kesimpulan dan Saran

5.1. Kesimpulan

Berdasarkan hasil pengukuran, pengujian, analisa dan pembahasan yang telah dilakukan, dapat disimpulkan bahwa:

1. Sistem deteksi kesehatan ayam petelur menggunakan YOLOv5 telah berhasil dibangun dan mampu mendeteksi objek ayam beserta fitur-fitur visualnya. Namun, tingkat confidence YOLOv5 mengalami penurunan ketika objek yang dideteksi berada pada jarak yang jauh, ukuran objek terlalu kecil seperti mata ayam, atau ketika ayam bergerak dan mengubah posisi.
2. Algoritma KNN telah diimplementasikan untuk mengklasifikasikan kondisi kesehatan ayam (sehat, sakit ringan, dan sakit parah) berdasarkan fitur yang diekstraksi dari hasil deteksi YOLOv5. Namun, integrasi YOLOv5 dengan KNN menunjukkan hasil yang kurang optimal karena karakteristik KNN yang bekerja dengan fitur numerik kurang sesuai dengan output deteksi visual YOLOv5, sehingga akurasi klasifikasi tidak mencapai 100% meskipun telah diterapkan mekanisme threshold confidence dan verifikasi ganda untuk hasil abnormal.
3. Analisis kinerja sistem YOLOv5-KNN menunjukkan bahwa meskipun kedua model bekerja dengan baik secara individual, penggabungan keduanya menghasilkan tantangan dalam hal akurasi prediksi kesehatan ayam petelur. Faktor-faktor yang mempengaruhi kinerja sistem meliputi jarak deteksi, ukuran objek, pergerakan ayam, dan keterbatasan kualitas kamera yang digunakan dalam pengambilan data.

5.2. Saran

Berdasarkan hasil penelitian dan kesimpulan yang telah dibuat, saran yang dapat diberikan untuk pengembangan penelitian selanjutnya adalah:

1. Mengganti algoritma klasifikasi KNN dengan metode klasifikasi lain yang lebih sesuai untuk bekerja dengan fitur gambar, karena KNN yang bekerja dengan membaca fitur atau kode numerik kurang cocok digabungkan dengan hasil deteksi visual YOLOv5, sehingga diharapkan dapat meningkatkan akurasi klasifikasi kesehatan ayam.
2. Melakukan pengumpulan data dan pengujian sistem secara langsung di kandang ayam petelur untuk mendapatkan kondisi yang sebenarnya, agar sistem dapat menyesuaikan dengan kondisi aktual seperti pencahayaan, jarak pengambilan gambar, dan pergerakan ayam di kandang.
3. Menggunakan perangkat kamera dengan kualitas dan spesifikasi yang lebih baik agar dapat menangkap detail yang lebih kecil seperti mata

ayam atau tanda-tanda penyakit dengan lebih jelas, sehingga nilai confidence deteksi YOLOv5 dapat meningkat.

4. Meningkatkan mekanisme verifikasi sistem untuk menangani kondisi ketika ayam bergerak atau berpindah posisi yang dapat mempengaruhi hasil deteksi, sehingga sistem dapat memberikan hasil klasifikasi yang lebih akurat dan konsisten.
5. Menambahkan lebih banyak variasi data pelatihan yang mencakup berbagai kondisi ayam pada jarak berbeda dan posisi berbeda, agar sistem lebih robust dalam mendeteksi kesehatan ayam meskipun objek yang dideteksi berukuran kecil atau berada pada jarak yang jauh.

Daftar Pustaka

[1] W. Wahyuni and A. Lestari, "Prevalensi Sakit dan Kematian Ayam Petelur (Studi Kasus di Peternakan Ayam Ras Petelur)," *Tarjih Tropical Livestock Journal*, vol. 2, no. 2, pp. 68–75, Dec. 2022, doi: 10.47030/trolija.v2i2.440.

[2] Y. Guo, P. Regmi, Y. Ding, R. B. Bist, and L. Chai, "Automatic detection of brown hens in cage-free houses with deep learning methods," *Poult Sci*, vol. 102, no. 8, Aug. 2023, doi: 10.1016/j.psj.2023.102784.

[3] L. Syafaah, A. Faruq, N. Setyawan, and M. I. Khair, "Sick and Dead Chicken Detection System Based on YOLO Algorithm," *Ingenierie des Systemes d'Information*, vol. 29, no. 5, pp. 1723–1729, Oct. 2024, doi: 10.18280/isi.290506.

[4] Y. Vita Via, F. Tri Anggraeny, and R. Andika Jorgie, "Seminar Nasional Informatika Bela Negara (SANTIKA) Penerapan Algoritma Case Based Reasoning dan K-Nearest Neighbor untuk Diagnosa Penyakit Ayam", [Online]. Available: https://www.cabi.org/isc

[5] A. Tasdelen and Y. Arslan, "Detection of high-risk diseases in poultry feces through transfer learning," *Engineering Science and Technology, an International Journal*, vol. 64, Apr. 2025, doi: 10.1016/j.jestch.2025.102002.

[6] A. J. Kalita, M. Subba, S. Adil, M. A. Wani, Y. A. Beigh, and M. Shafi, "Application of artificial intelligence and machine learning in poultry disease detection and diagnosis: A review," 2025, *Dr. Nasir Akbar Mir*. doi: 10.62310/liab.v5i1.155.

[7] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, "A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System," *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.

[8] R. Dulal *et al.*, "Automatic Cattle Identification using YOLOv5 and Mosaic Augmentation: A Comparative Analysis," 2022.

[9] A. Rohan, M. Saad Rafaq, J. Hasan, F. Asghar, A. Kashif Bashir, and T. Dottorini, "Application of deep learning for livestock behaviour recognition: A systematic literature review."

[10] A. Rohan, M. Saad Rafaq, J. Hasan, F. Asghar, A. Kashif Bashir, and T. Dottorini, "Application of deep learning for livestock behaviour recognition: A systematic literature review."

[11] R. Dulal *et al.*, "Automatic Cattle Identification using YOLOv5 and Mosaic Augmentation: A Comparative Analysis," 2022.

[12] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions," Oct. 2024, [Online]. Available:

http://arxiv.org/abs/2411.00201
[13] J. Terven, D. M. Córdova-Esparza, and J. A. Romero-González, “A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS,” Dec. 01, 2023, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/make5040083.
[14] N. Z. A. Ar Rahmah, Adianto, Rini Indarti, Zindhu Maulana Ahmad Putra, Edy Setiawan, and Afif Zuhri Arfianto, “Implementasi Deteksi Kelengkapan APD pada Hazardous Area menggunakan Metode YoloV5,” *Jurnal Elektronika dan Otomasi Industri*, vol. 11, no. 3, Oct. 2024, doi: 10.33795/elkolind.v11i3.5744.
[15] “wijaya et al (2024)”.
[16] R. Khanam, T. Asghar, and M. Hussain, “Comparative Performance Evaluation of YOLOv5, YOLOv8, and YOLOv11 for Solar Panel Defect Detection,” *Solar*, vol. 5, no. 1, Mar. 2025, doi: 10.3390/solar5010006.
[17] A. S. Geetha and M. Hussain, “A Comparative Analysis of YOLOv5, YOLOv8, and YOLOv10 in Kitchen Safety,” Jul. 2024, [Online]. Available: http://arxiv.org/abs/2407.20872
[18] N. Ma Muriyah, J. H. Sim, and A. Yulianto, “Evaluating YOLOv5 and YOLOv8: Advancements in Human Detection,” *Journal of Information Systems and Informatics*, vol. 6, no. 4, pp. 2999–3015, Dec. 2024, doi: 10.51519/journalisi.v6i4.944.
[19] I. Purwita Sary, E. Ucok Armin, S. Andromeda, E. Engineering, and U. Singaperbangsa Karawang, “Performance Comparison of YOLOv5 and YOLOv8 Architectures in Human Detection Using Aerial Images,” *Ultima Computing : Jurnal Sistem Komputer*, vol. 15, no. 1, 2023.
[20] A. S. Geetha and M. Hussain, “A Comparative Analysis of YOLOv5, YOLOv8, and YOLOv10 in Kitchen Safety,” Jul. 2024, [Online]. Available: http://arxiv.org/abs/2407.20872
[21] U. Nepal and H. Eslamiat, “Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs,” *Sensors*, vol. 22, no. 2, Jan. 2022, doi: 10.3390/s22020464.
[22] U. Nepal and H. Eslamiat, “Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs,” *Sensors*, vol. 22, no. 2, Jan. 2022, doi: 10.3390/s22020464.
[23] J. S. Murthy, G. M. Siddesh, W. C. Lai, B. D. Parameshachari, S. N. Patil, and K. L. Hemalatha, “ObjectDetect: A Real-Time Object Detection Framework for Advanced Driver Assistant Systems Using YOLOv5,” *Wirel Commun Mob Comput*, vol. 2022, 2022, doi: 10.1155/2022/9444360.
[24] J. S. Murthy, G. M. Siddesh, W. C. Lai, B. D. Parameshachari, S. N. Patil, and K. L. Hemalatha, “ObjectDetect: A Real-Time Object Detection Framework for Advanced Driver Assistant Systems Using YOLOv5,” *Wirel Commun Mob Comput*, vol. 2022, 2022, doi: 10.1155/2022/9444360.

[25] G. Sun, S. Wang, and J. Xie, “An Image Object Detection Model Based on Mixed Attention Mechanism Optimized YOLOv5,” *Electronics (Switzerland)*, vol. 12, no. 7, Apr. 2023, doi: 10.3390/electronics12071515.

[26] A. Akhtar, R. Ahmed, M. H. Yousaf, and S. A. Velastin, “Real-Time Motorbike Detection: AI on the Edge Perspective,” *Mathematics*, vol. 12, no. 7, Apr. 2024, doi: 10.3390/math12071103.

[27] H. Zhang, X. Zhou, Y. Shi, X. Guo, and H. Liu, “Object Detection Algorithm of Transmission Lines Based on Improved YOLOv5 Framework,” *J Sens*, vol. 2024, 2024, doi: 10.1155/2024/5977332.

[28] Q. Cheng, G. Yuan, D. Chen, B. Xu, E. Chen, and H. Zhou, “Transmission Lines Small-Target Detection Algorithm Research Based on YOLOv5,” *Applied Sciences (Switzerland)*, vol. 13, no. 16, Aug. 2023, doi: 10.3390/app13169386.

[29] A. Akhtar, R. Ahmed, M. H. Yousaf, and S. A. Velastin, “Real-Time Motorbike Detection: AI on the Edge Perspective,” *Mathematics*, vol. 12, no. 7, Apr. 2024, doi: 10.3390/math12071103.

[30] H. Peng, M. Liang, C. Yuan, and Y. Ma, “EDF-YOLOv5: An Improved Algorithm for Power Transmission Line Defect Detection Based on YOLOv5,” *Electronics (Switzerland)*, vol. 13, no. 1, Jan. 2024, doi: 10.3390/electronics13010148.

[31] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” Jul. 2022, [Online]. Available: http://arxiv.org/abs/2207.02696

[32] Q. Yu, Y. Han, Y. Han, X. Gao, and L. Zheng, “Enhancing YOLOv5 Performance for Small-Scale Corrosion Detection in Coastal Environments Using IoU-Based Loss Functions,” *J Mar Sci Eng*, vol. 12, no. 12, Dec. 2024, doi: 10.3390/jmse12122295.

[33] B. Yan, P. Fan, X. Lei, Z. Liu, and F. Yang, “A real-time apple targets detection method for picking robot based on improved YOLOv5,” *Remote Sens (Basel)*, vol. 13, no. 9, May 2021, doi: 10.3390/rs13091619.

[34] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, “A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System,” *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.

[35] R. K. Halder, M. N. Uddin, M. A. Uddin, S. Aryal, and A. Khraisat, “Enhancing K-nearest neighbor algorithm: a comprehensive review and performance analysis of modifications,” *J Big Data*, vol. 11, no. 1, Dec. 2024, doi: 10.1186/s40537-024-00973-y.

[36] A. A. Amer, S. D. Ravana, and R. A. A. Habeeb, “Enhanced Distance-based Weighted K-Nearest Neighbor Algorithm for Data Classification,” *KSII Transactions on Internet and Information Systems*, vol. 19, no. 4, pp.

1097–1121, 2025, doi: 10.3837/tiis.2025.04.003.
[37] A. İ. Çetin and A. H. Büyüklü, “A new approach to K-nearest neighbors distance metrics on sovereign country credit rating,” *Kuwait Journal of Science*, vol. 52, no. 1, Jan. 2025, doi: 10.1016/j.kjs.2024.100324.
[38] V. R. Tiwari, “Developments in KD Tree and KNN Searches,” 2023.
[39] W. Tao, G. Wang, Z. Sun, S. Xiao, Q. Wu, and M. Zhang, “Recognition Method for Broiler Sound Signals Based on Multi-Domain Sound Features and Classification Model,” *Sensors (Basel)*, vol. 22, no. 20, Oct. 2022, doi: 10.3390/s22207935.
[40] W. Tao, G. Wang, Z. Sun, S. Xiao, Q. Wu, and M. Zhang, “Recognition Method for Broiler Sound Signals Based on Multi-Domain Sound Features and Classification Model,” *Sensors (Basel)*, vol. 22, no. 20, Oct. 2022, doi: 10.3390/s22207935.
[41] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, “A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System,” *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.
[42] G. Ahmed, R. A. S. Malick, A. Akhunzada, S. Zahid, M. R. Sagri, and A. Gani, “An approach towards iot-based predictive service for early detection of diseases in poultry chickens,” *Sustainability (Switzerland)*, vol. 13, no. 23, Dec. 2021, doi: 10.3390/su132313396.
[43] M. A. A. A. Bakar, P. J. Ker, S. G. H. Tang, M. Z. Baharuddin, H. J. Lee, andA. R. Omar, “Translating conventional wisdom on chicken comb color into automated monitoring of disease-infected chicken using chromaticity- based machine learning models,” *Front Vet Sci*, vol. 10, 2023, doi: 10.3389/fvets.2023.1174700.
[44] V. Iglovikov, “Need for Speed: A Comprehensive Benchmark of JPEG Decoders in Python,” Jan. 2025, [Online]. Available: http://arxiv.org/abs/2501.13131
[45] S. Kastryulin, J. Zakirov, D. Prokopenko, and D. V. Dylov, “PyTorch Image Quality: Metrics for Image Quality Assessment,” Aug. 2022, [Online]. Available: http://arxiv.org/abs/2208.14818
[46] D. H. Kim, “A Study on an Effective Teaching of AI using Google Colab-Based DCGAN Deep Learning Model Building for Music Data Analysis and Genre Classification,” *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 11, no. 6, pp. 13–25, Mar. 2023, doi: 10.35940/ijrte.E7351.0311623.
[47] S. Samuel and D. Mietchen, “Computational reproducibility of Jupyter notebooks from biomedical publications,” *Gigascience*, vol. 13, 2024, doi: 10.1093/gigascience/giad113.
[48] J. Lin and P. P. W. Chen, “BakuFlow: A Streamlining Semi-Automatic Label Generation Tool,” Jun. 2025, [Online]. Available:

http://arxiv.org/abs/2506.09083
[49] C. Osborne, “Public-private funding models in open source software development: A case study on scikit-learn,” Apr. 2024, [Online]. Available: http://arxiv.org/abs/2404.06484
[50] A. Rohan, M. Saad Rafaq, J. Hasan, F. Asghar, A. Kashif Bashir, and T. Dottorini, “Application of deep learning for livestock behaviour recognition: A systematic literature review.”
[51] T. Y. Wei, T. H. Lin, and Y. C. Tsai, “Comb Color Analysis of Broilers Through the Video Surveillance System of a Poultry House,” *Revista Brasileira de Ciencia Avicola / Brazilian Journal of Poultry Science*, vol. 26, no. 1, 2024, doi: 10.1590/1806-9061-2023-1891.
[52] T. H. Kerbaa, A. Mezache, and H. Oudira, “Parameter Estimation in Radar K-Clutter plus Noise Based on Otsu’s Algorithm,” *Ingenierie des Systemes d’Information*, vol. 25, no. 3, pp. 295–302, Jun. 2020, doi: 10.18280/isi.250302.
[53] X. Yang, R. Bist, B. Paneru, and L. Chai, “Monitoring activity index and behaviors of cage-free hens with advanced deep learning technologies,” *Poult Sci*, vol. 103, no. 11, Nov. 2024, doi: 10.1016/j.psj.2024.104193.
[54] “2.2.1_ayam sehat”.
[55] H. Kim, W. Do Lee, H. K. Jang, M. Kang, and H. K. Kang, “The potential of non-movement behavior observation method for detection of sick broiler chickens,” *J Anim Sci Technol*, vol. 65, no. 2, pp. 441–458, 2023, doi: 10.5187/jast.2022.e105.
[56] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, “A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System,” *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.
[57] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, “A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System,” *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.
[58] P. K. Waliaula, E. G. Kiarie, and M. S. Diarra, “Predisposition factors and control strategies of avian pathogenic Escherichia coli in laying hens,” 2024, *Frontiers Media SA*. doi: 10.3389/fvets.2024.1474549.
[59] A. M. Janczak and A. B. Riber, “Review of rearing-related factors affecting the welfare of laying hens,” Apr. 23, 2015, *Oxford University Press*. doi: 10.3382/ps/pev123.
[60] A. Şekeroğlu, Y. E. Şentürk, B. Tainika, M. Duman, and A. Akyol, “The Impact of Laying Hen Age, Egg-Laying Time, Cage Tier, and Cage Direction on Egg Quality Traits in Hens in an Enriched Cage System,” *Revista Brasileira de Ciencia Avicola / Brazilian Journal of Poultry Science*, vol. 26, no. 2, 2024, doi: 10.1590/1806-9061-2024-1902.

[61] A. Şekeroğlu, Y. E. Şentürk, B. Tainika, M. Duman, and A. Akyol, “The Impact of Laying Hen Age, Egg-Laying Time, Cage Tier, and Cage Direction on Egg Quality Traits in Hens in an Enriched Cage System,” *Revista Brasileira de Ciencia Avicola / Brazilian Journal of Poultry Science*, vol. 26, no. 2, 2024, doi: 10.1590/1806-9061-2024-1902.

[62] A. M. Janczak and A. B. Riber, “Review of rearing-related factors affecting the welfare of laying hens,” Apr. 23, 2015, *Oxford University Press*. doi: 10.3382/ps/pev123.

[63] M. Musa, A. U. Mohammed, S. Musa, and L. M. Jabaka, “A K-Nearest Neighbor (KNN)-Algorithm in Poultry Diseases Monitoring System,” *International Journal of Science for Global Sustainability*, vol. 10, no. 3, pp. 47–56, Oct. 2024, doi: 10.57233/ijsgs.v10i3.696.

[64] T. H. Kerbaa, A. Mezache, and H. Oudira, “Parameter Estimation in Radar K-Clutter plus Noise Based on Otsu’s Algorithm,” *Ingenierie des Systemes d’Information*, vol. 25, no. 3, pp. 295–302, Jun. 2020, doi: 10.18280/isi.250302.

[65] J. H. So, S. Y. Joe, S. H. Hwang, S. J. Hong, and S. H. Lee, “Current advances in detection of abnormal egg: a review,” 2022, *Korean Society of Animal Sciences and Technology*. doi: 10.5187/jast.2022.e56.

[66] P. He *et al.*, “Research Progress in the Early Warning of Chicken Diseases by Monitoring Clinical Symptoms,” Jun. 01, 2022, *MDPI*. doi: 10.3390/app12115601.

[67] T. H. Kerbaa, A. Mezache, and H. Oudira, “Parameter Estimation in Radar K-Clutter plus Noise Based on Otsu’s Algorithm,” *Ingenierie des Systemes d’Information*, vol. 25, no. 3, pp. 295–302, Jun. 2020, doi: 10.18280/isi.250302.

[68] T. H. Kerbaa, A. Mezache, and H. Oudira, “Parameter Estimation in Radar K-Clutter plus Noise Based on Otsu’s Algorithm,” *Ingenierie des Systemes d’Information*, vol. 25, no. 3, pp. 295–302, Jun. 2020, doi: 10.18280/isi.250302.

Biodata

Nama	: Aldi Ananda Pakpahan
TTL	: Batam, 02 Agustus 2003
Agama	: Kristen
Alamat	: Sagulung sumber mulya, Blok A5 No 48
Email	: aldipakpahanjr@gmail.com
Riwayat Pendidikan	SMA/SMK : SMA Tunas Baru Batam
	SMP : SMP Negeri 21 Batam

Nama	: Marshanda Simanjuntak
TTL	: Bengkulu, 10 Maret 2004
Agama	: Islam
Alamat	: Legenda Malaka Aveneu Blok B4, No 1
Email	: marshandacaca0@gmail.com
Riwayat Pendidikan	SMA/SMK : Man 2 Bengkulu
	SMP : MTsN 2 Bengkulu

Lampiran

A. PROGRAM

1. Program Deteksi YOLOv5

```python
def _load_yolo_detector(self):
    """Load YOLO object detection model"""
    print(f"\n[LOADING] YOLO Object Detector: {self.config.yolo_model_path}...")
    try:
        self.yolo_detector = torch.hub.load(
            'ultralytics/yolov5',
            'custom',
            path=self.config.yolo_model_path,
            force_reload=False
        )
        self.yolo_detector.conf = self.config.yolo_confidence_threshold
        self.yolo_detector.to(self.config.device_type).eval()
        print(f"  ✓ YOLO detector loaded successfully")
        print(f"  → Model device: {self.config.device_type}")
        print(f"  → Confidence threshold: {self.config.yolo_confidence_threshold}")
    except Exception as e:
        print(f"  ✗ Failed to load YOLO detector: {e}")
        self.yolo_detector = None
```

```python
def process_inference_frame(self, frame_data):
    detection_results = []
    inference_start = time.time()

    if self.yolo_detector is None:
        return detection_results, None

    # YOLO inference phase
    with torch.no_grad():
        yolo_results = self.yolo_detector(frame_data, size=416)

    yolo_predictions = yolo_results.pandas().xyxy[0]

    for detection_index, detection_row in yolo_predictions.iterrows():
        detection_class = int(detection_row['class'])
        detection_label = detection_row['name'].lower()
        yolo_confidence = float(detection_row['confidence'])

        # Filter to known classes (0=normal, 1=abnormal)
        if detection_class not in [0, 1]:
            continue

        # Extract bounding box coordinates
        bbox_x1, bbox_y1 = int(detection_row['xmin']), int(detection_row['ymin'])
        bbox_x2, bbox_y2 = int(detection_row['xmax']), int(detection_row['ymax'])
```

2. Program Training KNN

```python
def _load_knn_classifier(self):
    """Load KNN disease classification model"""
    print(f"\n[LOADING] KNN Disease Classifier...")
    try:
        models_base = Path(self.config.knn_models_dir)
        knn_path = models_base / self.config.knn_model_file
        scaler_path = models_base / self.config.knn_scaler_file
        encoder_path = models_base / self.config.knn_encoder_file

        self.knn_classifier = joblib.load(str(knn_path))
        self.feature_scaler = joblib.load(str(scaler_path))
        self.label_encoder = joblib.load(str(encoder_path))

        print(f"  ✓ KNN classifier loaded successfully")
        print(f"  → Disease classes: {list(self.label_encoder.classes_)}")
        print(f"  → Feature dimension: {self.feature_scaler.n_features_in_}")
    except Exception as e:
        print(f"  X Failed to load KNN classifier: {e}")
```

```python
if detection_label == "abnormal" and self.knn_classifier:
    roi_region = frame_data[bbox_y1:bbox_y2, bbox_x1:bbox_x2]

    if roi_region.size > 0:
        extracted_features = extract_features_from_roi(roi_region)

        if (extracted_features is not None and
            extracted_features.shape[1] == self.config.feature_dimension_expected):

            normalized_features = self.feature_scaler.transform(extracted_features)
            predicted_class_idx = self.knn_classifier.predict(normalized_features)[0]
            classified_disease = self.label_encoder.inverse_transform([predicted_class_idx])[0]

            # Get probability predictions
            class_probabilities = self.knn_classifier.predict_proba(normalized_features)[0]
            disease_confidence = float(class_probabilities[predicted_class_idx])
```

3. Program YOLOv5-KNN

```python
205         # YOLO inference phase
206         with torch.no_grad():
207             yolo_results = self.yolo_detector(frame_data, size=416)
208
209         yolo_predictions = yolo_results.pandas().xyxy[0]
210
211         for detection_index, detection_row in yolo_predictions.iterrows():
212             detection_class = int(detection_row['class'])
213             detection_label = detection_row['name'].lower()
214             yolo_confidence = float(detection_row['confidence'])
215
216             # Filter to known classes (0=normal, 1=abnormal)
217             if detection_class not in [0, 1]:
218                 continue
219
220             # Extract bounding box coordinates
221             bbox_x1, bbox_y1 = int(detection_row['xmin']), int(detection_row['ymin'])
222             bbox_x2, bbox_y2 = int(detection_row['xmax']), int(detection_row['ymax'])
223
224             # Initialize disease classification fields
225             classified_disease = "N/A"
226             disease_confidence = 0.0
227             top_disease_predictions = []
```

```
# KNN classification for abnormal detections
if detection_label == "abnormal" and self.knn_classifier:
    roi_region = frame_data[bbox_y1:bbox_y2, bbox_x1:bbox_x2]

    if roi_region.size > 0:
        extracted_features = extract_features_from_roi(roi_region)

        if (extracted_features is not None and
            extracted_features.shape[1] == self.config.feature_dimension_expected):

            normalized_features = self.feature_scaler.transform(extracted_features)
            predicted_class_idx = self.knn_classifier.predict(normalized_features)[0]
            classified_disease = self.label_encoder.inverse_transform([predicted_class_idx])[0]

            # Get probability predictions
            class_probabilities = self.knn_classifier.predict_proba(normalized_features)[0]
            disease_confidence = float(class_probabilities[predicted_class_idx])

            # Track top-3 predictions
            top3_indices = np.argsort(class_probabilities)[-3:][::-1]
            top_disease_predictions = [
                {
                    'disease': self.label_encoder.inverse_transform([idx])[0],
                    'confidence': float(class_probabilities[idx])
                }
                for idx in top3_indices
            ]

            self.performance_metrics['disease_counts'][classified_disease] += 1
            self.performance_metrics['confidence_history'].append(disease_confidence)
```

```
        # Update detection counters
        self.performance_metrics['total_inference_frames'] += 1
        if detection_label == 'normal':
            self.performance_metrics['healthy_detections'] += 1
        elif detection_label == 'abnormal':
            self.performance_metrics['abnormal_detections'] += 1

        # Assemble detection result
        inference_latency = (time.time() - inference_start) * 1000
        self.performance_metrics['inference_times'].append(inference_latency)

        detection_results.append({
            'detection_id': self.performance_metrics['total_inference_frames'],
            'detection_class': detection_label,
            'classified_disease': classified_disease,
            'yolo_confidence': yolo_confidence,
            'knn_confidence': disease_confidence,
            'bounding_box': [bbox_x1, bbox_y1, bbox_x2, bbox_y2],
            'top_predictions': top_disease_predictions,
            'inference_type': 'REALTIME',
            'processing_latency_ms': inference_latency,
            'timestamp': datetime.now()
        })

    return detection_results, yolo_results
```

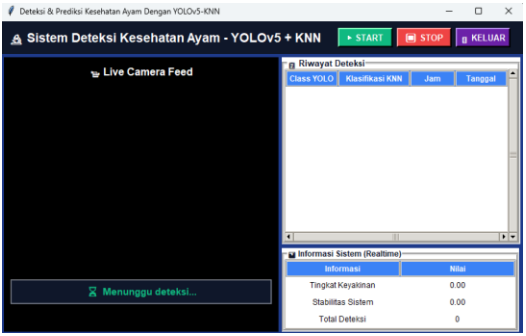
4. Program Ekstrak Fitur

```python
def extract_features_from_roi(roi_image):
    try:
        if roi_image is None or roi_image.size == 0:
            return None

        # Standardize image size
        standardized_roi = cv2.resize(roi_image, (128, 128))
        rotation_features = []

        # Process each rotation for invariance
        for rotation_angle in [0, 90, 180, 270]:
            if rotation_angle == 0:
                rotated_image = standardized_roi
            elif rotation_angle == 90:
                rotated_image = cv2.rotate(standardized_roi, cv2.ROTATE_90_CLOCKWISE)
            elif rotation_angle == 180:
                rotated_image = cv2.rotate(standardized_roi, cv2.ROTATE_180)
            else:  # 270
                rotated_image = cv2.rotate(standardized_roi, cv2.ROTATE_90_COUNTERCLOCKWISE)

            frame_features = []

            # ===== COLOR STATISTICS (6 features) =====
            for channel_idx in range(3):
                channel_data = rotated_image[:, :, channel_idx]
                frame_features.append(np.mean(channel_data))
                frame_features.append(np.std(channel_data))

            # ===== COLOR HISTOGRAMS (24 features) =====
            for channel_idx in range(3):
                histogram_bins = cv2.calcHist(
                    [rotated_image],
                    [channel_idx],
                    None,
                    [8],
                    [0, 256]
                )
                frame_features.extend(histogram_bins.flatten().tolist())
```

```python
            # ===== EDGE DETECTION FEATURES (5 features) =====
            sobel_x = cv2.Sobel(gray_image, cv2.CV_64F, 1, 0, ksize=3)
            sobel_y = cv2.Sobel(gray_image, cv2.CV_64F, 0, 1, ksize=3)
            edge_magnitude = np.sqrt(sobel_x**2 + sobel_y**2)

            frame_features.extend([
                np.mean(edge_magnitude),
                np.std(edge_magnitude),
                np.min(edge_magnitude),
                np.max(edge_magnitude),
                np.median(edge_magnitude)
            ])

            # Total per rotation: 6 + 24 + 4 + 16 + 5 = 55 features
            rotation_features.append(frame_features)

        # Average across rotations for invariance
        aggregated_features = np.mean(rotation_features, axis=0)
        feature_vector = aggregated_features.reshape(1, -1)

        # Validate feature dimension
        if feature_vector.shape[1] != 55:
            print(f"[WARNING] Feature vector dimension: {feature_vector.shape[1]} (expected 55)")

        return feature_vector

    except Exception as e:
        print(f"[ERROR] Feature extraction failed: {e}")
        import traceback
        traceback.print_exc()
        return None


# Alias for backward compatibility
extract_features = extract_features_from_roi
```

B. Dokumentasi

1. GUI



2. Alat

3. Implementasi

C. Berikut ini adalah link drive semua program dan data gambar:
https://drive.google.com/drive/folders/1OcU9vbHeR2iIzaaNpG9O0SrWHwtr1YkI?usp=drive_link

FORMULIR LOOGBOOK BIMBINGAN DAN PENGAJUAN SIDANG TUGAS AKHIR

Nama : Marshanda Simanjuntak
NIM : 4212201035
Pembimbing I : M. Naufal Airlangga Diputra,S.Pd., M.P.H
Judul : Deteksi & Prediksi Kesehatan Ayam dengan YOLO - K-Nearest Neighbor

No	Hari/Tgl	Rincian Kegiatan	TTD Pembimbing
1	Rabu, 10-09-2025	Membahas data kaki ayam YOLO & KNN dalam deteksi dan klasifikasi	
2	Jum'at, 19-09-2025	Mencari data gambar kesehatan ayam dibeberapa website (Roboflow)	
3	Senin, 29-09-2025	Proses labeling data melalui Roboflow	
4	Senin, 06-10-2025	Pelatihan model deteksi (YOLO)	
5	Rabu, 15-10-2025	Proses menyelesaikan fitur gambar kesehatan ayam untuk diuji oleh KNN	
6	Jumat, 24-10-2025	Pelatihan model Prediksi (KNN)	
7	Senin, 02-11-2025	membuat program yang menggabungkan YOLO dengan metode KNN	
8	Rabu, 19-11-2025	Proses pembuatan website deteksi	
9	Senin, 08-12-2025	integrasi model GUI (Tampilan Data)	
10	Kamis, 18-12-2025	Pengumpulan laporan akhir	

Berdasarkan hasil bimbingan yang telah dilaksanakan selama 4 bulan dan telah disetujui oleh dosen pembimbing, maka dengan ini saya mengajukan diri sebagai peserta Tugas Akhir.

Batam, 29 September 2025
Peserta

Marshanda Simanjuntak
NIM: 4212201035