

Article

LeafScan: Aplikasi Cerdas untuk Deteksi Penyakit pada Daun Jagung

Rasyad Bimasatya¹, Mario Valerian Rante Ta'dung¹, Dhea Ayu Eka Natalia Pratiwi¹, Rahmatia¹, Evan Pandu Nata¹, A M Fauzan Baihaqi T¹, and A. Afif Alhaq¹

¹ Sistem Informasi, Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin

* Emasih Korespondensi: bimasatyar22h@student.unhas.ac.id

† Alamat Saat Ini: Jl. Perintis Kemerdekaan Km.10 Tamalanrea, Makassar, Sulawesi Selatan, Indonesia.

‡ Penulis-penulis ini berkontribusi secara setara terhadap karya ini.

Abstract: Penyakit daun jagung adalah salah satu ancaman utama dalam pertanian yang dapat menurunkan hasil panen secara signifikan. Deteksi dini terhadap penyakit ini penting untuk mencegah penyebaran dan mengurangi kerugian. Namun, metode deteksi manual sering kali memakan waktu dan tidak cukup akurat. Untuk itu, penelitian ini mengembangkan aplikasi *LeafScan* yang memanfaatkan teknologi *deep learning* untuk deteksi penyakit daun jagung secara otomatis dan cepat. Model deteksi objek yang digunakan adalah YOLOv8, yang terkenal dengan efisiensi dan akurasi tinggi dalam mendeteksi objek. Untuk mengatasi tantangan dalam mendeteksi objek kecil, diterapkan teknik *Slicing Aided Hyper Inference* (SAHI), yang membagi gambar menjadi potongan-potongan kecil tumpang tindih, meningkatkan presisi deteksi dan menggabungkan hasilnya menggunakan teknik *Non-Maximum Suppression* (NMS). Hasil penelitian menunjukkan peningkatan dalam *precision*, *recall*, dan *mean Average Precision* (mAP), khususnya untuk penyakit *blight*, meskipun deteksi untuk penyakit *rust* perlu ditingkatkan. Aplikasi ini, yang dikembangkan dengan antarmuka *Flutter*, diharapkan dapat membantu petani mendeteksi penyakit dengan lebih cepat dan akurat.

Keywords: Deteksi Penyakit Daun Jagung; YOLOv8; Deep Learning; Slicing Aided Hyper Inference (SAHI); Deteksi Objek; Flutter;

1. Pendahuluan

Jagung merupakan salah satu komoditas pangan utama yang memiliki peran penting dalam sektor pertanian, terutama di Indonesia, di mana ia menjadi sumber makanan dan pakan ternak. Namun, kesehatan tanaman jagung sangat bergantung pada kondisi daun, yang merupakan organ utama dalam proses fotosintesis. Saat ini, serangan penyakit seperti bercak daun, karat daun, dan hawar daun sering menjadi ancaman serius yang dapat menurunkan kualitas serta kuantitas hasil panen. Studi menunjukkan bahwa teknologi pemrosesan citra seperti Convolutional Neural Network (CNN) dapat mendeteksi penyakit pada daun jagung dengan tingkat akurasi hingga 94% [1].

Petani menghadapi tantangan besar untuk menemukan penyakit pada tanaman jagung. Penyakit dapat mengurangi hasil panen secara signifikan jika tidak diidentifikasi dengan benar. Oleh karena itu, sangat penting untuk mengembangkan aplikasi yang menggunakan pemrosesan gambar dan kecerdasan buatan (AI) untuk mendeteksi penyakit daun jagung. Tujuan aplikasi ini adalah untuk menawarkan cara yang efektif untuk melacak kesehatan tanaman, mengidentifikasi penyakit, dan memberikan saran perawatan yang tepat. Selain itu, sebuah studi mengembangkan sistem pakar berbasis Android yang dapat mendeteksi penyakit dan hama pada tanaman jagung, menunjukkan betapa pentingnya alat untuk pendeksi dini penyakit dan hama pada tanaman jagung bagi petani [2].

Adanya aplikasi LeafScan diharapkan akan membantu petani menemukan penyakit daun jagung dengan cepat dan akurat. Dengan menggunakan teknologi AI, aplikasi ini

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Journal Not Specified* 2024, 1, 0. <https://doi.org/>

Received:

Revised:

Accepted:

Published:

Copyright: © 2024 by the authors. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

dapat menemukan pola-pola visual yang menunjukkan penyakit pada gambar daun jagung. Hal ini akan meningkatkan akurasi diagnosis dan mempercepat proses deteksi, yang memungkinkan tindakan pencegahan dilakukan lebih awal. Dari penelitian sebelumnya menunjukkan bahwa model berbasis CNN dapat mengidentifikasi penyakit pada daun jagung dengan akurasi 93%. Hasil tersebut dapat mendukung adanya pengembangan aplikasi ini [3].

Penelitian ini sangat penting karena kondisi lapangan saat ini, di mana petani sering menghadapi masalah dalam menemukan dan mengatasi penyakit tanaman. Aplikasi ini diharapkan dapat menurunkan kesalahan diagnosis dan meningkatkan produktivitas pertanian jagung secara keseluruhan. Studi ini mengevaluasi seberapa baik aplikasi LeafScan mengidentifikasi penyakit pada tanaman jagung dengan lebih akurat dan cepat.

Penelitian ini akan mengukur seberapa akurat diagnosis yang dapat dicapai dengan metode manual dibandingkan dengan aplikasi LeafScan, dan bagaimana aplikasi ini dapat membantu petani menemukan penyakit daun jagung dengan cepat dan akurat. Penelitian ini berfokus pada pembuatan alat yang bermanfaat dan relevan bagi petani dengan tujuan peluncuran aplikasi yang jelas dan terukur.

Penelitian ini mencakup pembuatan aplikasi yang diperuntukan kepada petani, pelajar dan juga mahasiswa pertanian. Oleh karena itu, hasilnya diharapkan dapat membantu sektor pertanian di Indonesia melalui peningkatan pengetahuan dan kemampuan petani untuk menemukan dan mengobati penyakit pada tanaman jagung.

2. Metode

Aplikasi ini menggunakan pendekatan berbasis *deep learning* untuk mendeteksi penyakit pada daun jagung dengan cepat dan akurat melalui pemrosesan gambar. Metode penelitian ini melibatkan beberapa tahap, yaitu pengumpulan data, penjelasan data, pemilihan algoritma atau model, prosedur pengujian, serta evaluasi performa model. Berikut adalah rincian metode yang digunakan:

2.1. Pengumpulan Data

Data untuk penelitian ini diperoleh dari platform Kaggle, yang merupakan salah satu sumber terkemuka untuk berbagai dataset dalam berbagai bidang. Dataset yang digunakan dalam penelitian ini berisi 4.188 gambar daun jagung dengan berbagai kondisi kesehatan dan jenis penyakit. Setiap gambar telah diberi label sesuai dengan jenis penyakitnya, sehingga data ini dapat mendukung keperluan pelatihan dan pengujian model deteksi penyakit berbasis *deep learning*.

2.2. Penjelasan Data

Dataset yang digunakan dalam penelitian ini terdiri dari gambar daun jagung dengan berbagai kondisi kesehatan, yang dikelompokkan ke dalam empat kategori utama. Data dikumpulkan dari platform Kaggle dan mencakup total 4.188 gambar berformat JPG yang terbagi dalam kategori sebagai berikut:

Berikut adalah penjelasan mengenai masing-masing kelas dalam dataset ini:

- *Common Rust*: Kelas ini mencakup gambar daun jagung yang terinfeksi oleh penyakit karat daun, yang disebabkan oleh jamur *Puccinia sorghi* [4]. Penyakit ini mengakibatkan bercak-bercak coklat kemerahan di permukaan daun, dan daun yang terinfeksi biasanya menunjukkan tanda-tanda penguningan pada bagian tengah daun.
- *Gray Leaf Spot*: Kelas ini berisi gambar daun jagung yang terkena penyakit bercak daun abu-abu, yang disebabkan oleh jamur *Cercospora zeae-maydis* [5]. Ciri-ciri daun yang terinfeksi adalah munculnya bercak abu-abu yang berujung hitam pada daun, yang dapat mengurangi efisiensi fotosintesis pada tanaman.
- *Blight*: Kelas ini terdiri dari gambar daun yang terinfeksi oleh penyakit hawar daun, yang dapat disebabkan oleh berbagai patogen seperti jamur atau bakteri. Penyakit ini menyebabkan daun jagung menguning, mati, dan mengering, yang dapat merusak jaringan daun secara luas [6].

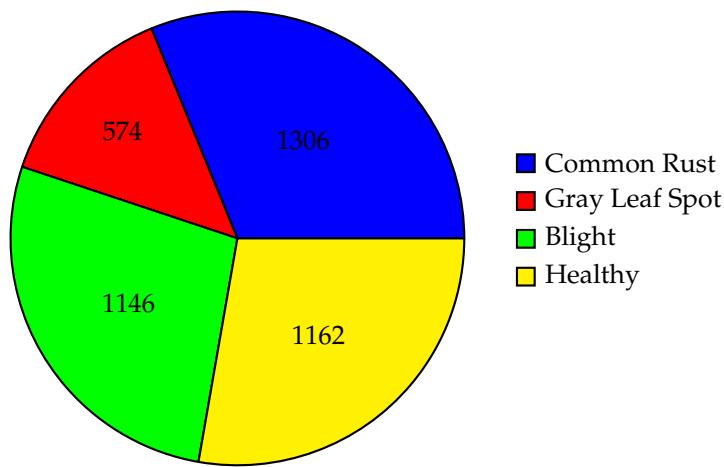


Figure 1. Diagram Lingkaran untuk Jumlah Data pada Setiap Kategori

- *Healthy*: Kelas ini mencakup gambar daun jagung yang tidak terinfeksi penyakit apapun, yang terlihat segar dan berwarna hijau normal. Daun jagung sehat memiliki tekstur yang tidak rusak dan tidak menunjukkan tanda-tanda kerusakan atau infeksi.

2.3. Algoritma atau Model

• YOLOv8:

YOLOv8 yang dirilis pada Januari 2023 oleh Ultralytics, menawarkan berbagai versi skala untuk deteksi objek, mulai dari YOLOv8n (nano) hingga YOLOv8x (extra-large). Arsitekturnya mirip dengan YOLOv5, namun mengganti CSPLayer dengan modul C2f (cross-stage partial bottleneck dengan dua konvolusi) untuk meningkatkan akurasi deteksi. YOLOv8 mengadopsi model tanpa anchor dengan head yang dipisah untuk menangani tugas objectness, classification, dan regression secara mandiri, yang berkontribusi pada peningkatan akurasi model secara keseluruhan. Dengan kecepatan dan efisiensi yang tinggi, YOLOv8 menjadi model yang sangat andal dalam deteksi objek [15]. Berikut penjelasan lengkap dari arsitektur YoloV8.

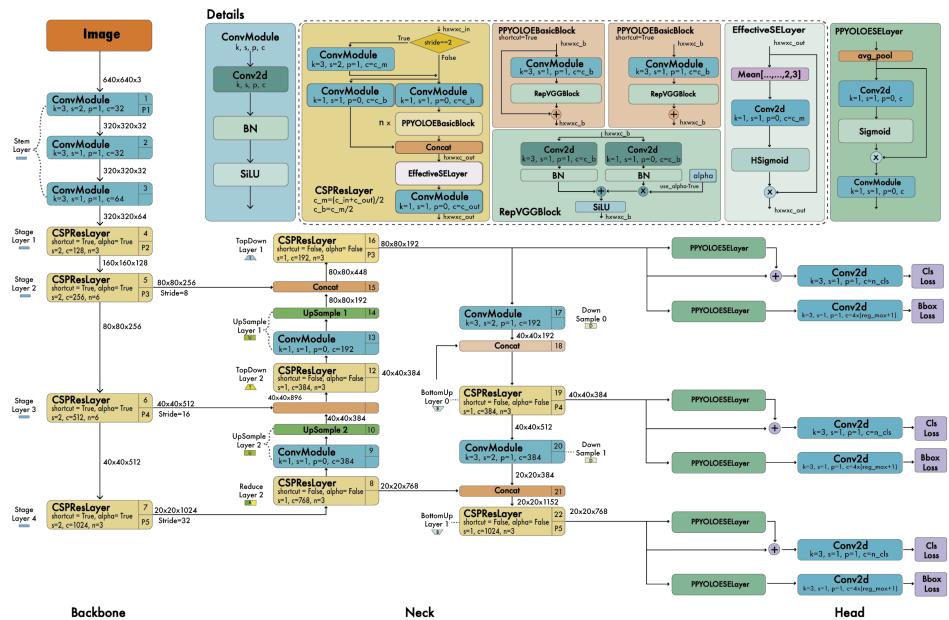


Figure 2. Arsitektur YoloV8 [15]

1. Backbone

Backbone merupakan bagian pertama dalam arsitektur YOLOv8 yang bertanggung jawab untuk mengekstraksi fitur utama dari input gambar. Proses ini mencakup penurunan resolusi gambar secara bertahap untuk menghasilkan representasi fitur yang lebih mendalam dan kompleks, sambil meningkatkan jumlah channel untuk memperkaya informasi. Backbone terdiri dari dua bagian utama, yaitu Stem Layer dan Stage Layers.

Pada bagian Stem Layer, YOLOv8 memanfaatkan tiga lapisan ConvModule yang dirancang untuk memproses gambar awal yang berukuran 640x640x3. Setiap ConvModule terdiri dari operasi konvolusi 2D untuk menangkap pola fitur dasar, batch normalization untuk menstabilkan distribusi output, dan fungsi aktivasi SiLU untuk menambahkan non-linearitas. Lapisan ini secara bertahap mengurangi resolusi gambar sambil meningkatkan jumlah channel fitur hingga menghasilkan keluaran berukuran 320x320x64.

Setelah melewati Stem Layer, Backbone melanjutkan proses melalui empat Stage Layers. Pada Stage Layer 1, model menggunakan CSPResLayer untuk menangkap fitur pada resolusi tinggi dengan jumlah channel sebesar 128, menghasilkan keluaran berukuran 160x160x128. Di tahap berikutnya, yaitu Stage Layer 2, resolusi gambar kembali dikurangi menjadi 80x80, sementara jumlah channel meningkat menjadi 256. Proses ini terus berlangsung hingga Stage Layer 4, di mana ukuran keluaran akhir Backbone adalah 20x20x1024. Progresi ini memastikan bahwa Backbone menangkap fitur dari berbagai tingkat resolusi, mulai dari detail lokal hingga pola global yang kompleks.

2. Neck

Setelah fitur diekstraksi oleh Backbone, bagian Neck bertugas untuk menggabungkan informasi fitur dari berbagai tingkat resolusi. Hal ini bertujuan untuk memperkuat kemampuan deteksi multi-skala, sehingga YOLOv8 dapat mendeteksi objek kecil, sedang, maupun besar dengan akurat. Neck dirancang dengan dua jalur utama, yaitu Top-Down Path dan Bottom-Up Path.

Pada jalur Top-Down Path, fitur yang dihasilkan oleh Backbone pada resolusi rendah, seperti 20x20x1024, di-upsample menjadi resolusi yang lebih tinggi, misalnya 40x40x512. Proses upsampling ini memungkinkan penggabungan fitur dari resolusi rendah dengan fitur dari Backbone yang memiliki resolusi lebih tinggi, seperti 40x40x512. Setelah resolusi fitur diselaraskan, operasi concatenation dilakukan untuk mengintegrasikan informasi dari kedua skala. Hasil penggabungan ini kemudian diproses lebih lanjut menggunakan ConvModule untuk menyelaraskan jumlah channel dan memperbaiki kualitas fitur. Sebagai langkah tambahan, CSPResLayer digunakan untuk menonjolkan fitur yang penting, dengan memanfaatkan mekanisme residual yang membantu mempertahankan informasi awal.

Selain jalur Top-Down Path, Neck juga mencakup jalur Bottom-Up Path yang bekerja dengan menyusun ulang fitur ke resolusi yang lebih rendah. Tujuan jalur ini adalah memastikan bahwa fitur dari skala besar dan kecil diproses bersama-sama untuk mendukung deteksi objek dengan ukuran yang bervariasi.

3. Head

Bagian terakhir dari YOLOv8 adalah Head, yang bertugas menghasilkan prediksi akhir, termasuk regresi bounding box, klasifikasi objek, dan confidence score. Pada bagian ini, YOLOv8 menggunakan PPYOLOE Layer yang didukung oleh blok RepVGGBlock. PPYOLOE Layer bertanggung jawab memproses fitur dari Neck dengan efisiensi tinggi, sekaligus menyelaraskan fitur untuk mempersiapkan tahap akhir prediksi.

Pada tahap prediksi, Head menggunakan dua lapisan Conv2D. Lapisan pertama digunakan untuk menghasilkan prediksi klasifikasi objek, sementara lapisan kedua bertugas melakukan regresi bounding box. Setiap fitur dari berba-

gai resolusi, seperti P3, P4, dan P5, diproses secara terpisah untuk memastikan bahwa prediksi dilakukan dengan mempertimbangkan skala resolusi yang sesuai. Desain ini memungkinkan YOLOv8 mendekripsi objek dari berbagai ukuran dengan lebih akurat. Salah satu keunggulan Head YOLOv8 adalah pendekatan anchor-free-nya, yang menyederhanakan proses pelatihan dan meningkatkan efisiensi.

155
156
157
158
160

- **Penggunaan SAHI dalam Model**

Slicing Aided Hyper Inference (SAHI) menggunakan teknik pemotongan selama proses inferensi [8]. Gambar asli I dibagi menjadi beberapa potongan tumpang tindih dengan ukuran $M \times N$, yang disebut $P_{I1}, P_{I2}, \dots, P_{In}$. Setiap potongan kemudian diubah ukurannya dengan menjaga rasio aspek gambar. Setelah itu, deteksi objek dilakukan pada setiap potongan secara terpisah. Jika diperlukan, full-inference (FI) juga bisa dilakukan dengan gambar asli untuk mendekripsi objek yang lebih besar. Hasil prediksi dari potongan-potongan ini dan, jika ada, hasil FI digabungkan kembali ke dalam gambar asli menggunakan teknik Non-Maximum Suppression (NMS). Pada NMS, kotak yang memiliki nilai Intersection over Union (IoU) lebih besar dari ambang batas yang ditentukan T_m akan dipadankan, dan deteksi dengan probabilitas di bawah ambang batas T_d akan dihapus.

161
162
163
164
165
166
167
168
169
170
171
172

2.4. Pra-Pelatihan (Prosedur)

Aplikasi Roboflow digunakan untuk melakukan anotasi pada data untuk menandai semua jenis penyakit berdasarkan 3 kategori yang telah ditentukan. Kemudian, data yang telah dianotasi diproses terlebih dahulu sebelum digunakan dalam pelatihan model. Berikut adalah prosesnya:

173
174
175
176
177

- **Pra-pemrosesan**

- **Auto-Orient**

Pengaturan ini secara otomatis memperbaiki orientasi gambar berdasarkan metadata, memastikan semua gambar memiliki orientasi yang konsisten.

- **Resize**

Setiap gambar diubah ukurannya menjadi 640×640 piksel. Ukuran ini diperlukan untuk melatih model YOLOv8.

178
179
180
181
182
183
184

- **Pembagian Data**

Data yang telah melalui pra-pemrosesan akan dibagi menjadi tiga kategori, yaitu data latih, data uji, dan data evaluasi, dengan proporsi masing-masing sebesar 80:10:10. Pembagian ini dilakukan agar model dapat dilatih, divalidasi, dan diuji secara menyeluruh. Pembagian data tersebut adalah sebagai berikut:

185
186
187
188
189

- **Data Latih (80%):** Digunakan untuk melatih model agar dapat mengenali pola dan karakteristik pada gambar daun jagung berdasarkan kategori penyakitnya.
- **Data Uji (10%):** Digunakan untuk mengevaluasi performa model selama proses pengembangan guna memastikan model tidak mengalami overfitting atau underfitting.
- **Data Evaluasi (10%):** Digunakan sebagai tes akhir untuk mengukur performa model secara independen pada data yang belum pernah dilihat sebelumnya, memastikan akurasi prediksi yang baik pada data nyata.

190
191
192
193
194
195
196
197

- **Augmentasi**

Augmentasi gambar hanya dilakukan pada data kategori latih (train). Langkah ini diambil untuk meningkatkan jumlah dan variasi data latih, sehingga model dapat belajar dari lebih banyak contoh yang berbeda. Dengan cara ini, model dapat menjadi lebih robust dan lebih mampu mengenali pola pada data yang lebih beragam, tanpa perlu mengubah data uji atau evaluasi yang harus tetap representatif terhadap kondisi data yang sesungguhnya.

198
199
200
201
202
203
204

- **Hue**
Menyesuaikan hue setiap gambar secara acak dalam rentang -15 hingga +15 derajat. Ini mensimulasikan variasi warna kecil untuk membuat model lebih kuat terhadap perubahan warna.205
206
207
208
- **Saturation**
Mengubah saturasi setiap gambar dalam rentang -30% hingga +30%. Ini mengubah intensitas warna, membantu model beradaptasi dengan gambar yang memiliki kecerahan warna yang berbeda.209
210
211
212
- **Brightness**
Mengubah kecerahan setiap gambar dengan menyesuaikannya secara acak antara -23% hingga +23%. Ini membantu model belajar menangani gambar dengan kondisi pencahayaan yang berbeda.213
214
215
216
- **Exposure**
Mengubah eksposur setiap gambar dalam rentang -10% hingga +10%, mensimulasikan berbagai lingkungan pencahayaan untuk meningkatkan adaptabilitas model terhadap tingkat eksposur yang berbeda.217
218
219
220
- **Blur**
Menambahkan efek blur dengan radius maksimum 2.2 piksel. Ini meniru gambar yang tidak fokus atau blur karena gerakan, yang dapat membuat model lebih tahan terhadap sedikit keburaman pada data input.221
222
223
224
- **Bounding Box: Flip (Horizontal)**
Pengaturan ini membalik gambar dan anotasi bounding box secara horizontal. Augmentasi ini memberikan pandangan cermin objek, meningkatkan kemampuan model untuk mengenali fitur dari berbagai perspektif.225
226
227
228

2.5. Metrik Evaluasi

229

- **Box Loss**
Model YOLO yang dikembangkan oleh Ultralytics menggunakan Complete Intersection Over Union (CIoU) untuk menilai seberapa baik prediksi bounding box cocok dengan bounding box target dalam tugas deteksi objek[14]. Berikut adalah Persamaan CiOU:230
231
232
233
234

$$\text{CiOU}(A, B) = 1 - \text{IoU}(A, B) + \frac{\rho^2(\mathbf{b}, \mathbf{b}^g)}{c^2} + \alpha v$$

Keterangan:

- $\text{IoU}(A, B)$ adalah Intersection over Union antara bounding box A dan B .235
236
- $\rho(\mathbf{b}, \mathbf{b}^g)$ adalah jarak Euklidean antara pusat dari bounding box prediksi \mathbf{b} dan bounding box target \mathbf{b}^g .237
238
- c adalah diagonal dari kotak yang mengelilingi kedua bounding box.239
- α dan v adalah parameter penyeimbang untuk aspek rasio.240

- **Class Loss**

Classification Loss (class loss atau cls loss) dalam YOLOv8 mengukur seberapa akurat model dalam mengklasifikasikan objek yang terdeteksi ke dalam kelas yang benar. Ini dilakukan dengan membandingkan probabilitas kelas yang diprediksi model dengan kelas yang sebenarnya (ground truth). Cross-Entropy Loss digunakan dalam perhitungan class loss[16]. Berikut adalah Persamaan Cross-Entropy Loss:241
242
243
244
245
246

$$L_{\text{cls}} = - \sum_{c=1}^C y_c \log(p_c)$$

Keterangan:

- C adalah jumlah kelas dalam dataset (misalnya, jumlah kelas objek yang perlu dideteksi).247
248
249

- y_c adalah label kelas target (nilai 1 jika objek termasuk dalam kelas c , atau 0 jika tidak).

- p_c adalah probabilitas prediksi untuk kelas c , yang dihitung oleh model.

- **Precision**

Precision mengukur seberapa banyak prediksi positif model yang benar. Ini dihitung dengan membandingkan jumlah deteksi yang benar dengan total deteksi yang dilakukan. Precision memberikan informasi tentang seberapa tepat model dalam memprediksi objek, yang penting untuk mengevaluasi apakah model menghasilkan banyak prediksi yang salah atau tidak[13]. Berikut adalah Persamaan Precision:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Keterangan:

- TP adalah jumlah *True Positives* (prediksi benar yang cocok dengan ground truth).
- FP adalah jumlah *False Positives* (prediksi salah).

- **Recall**

Recall mengukur seberapa banyak objek yang benar-benar ada dalam gambar yang berhasil dideteksi oleh model. Recall memberikan informasi tentang seberapa baik model dalam menemukan objek yang ada dalam gambar, yang penting untuk mengevaluasi kemampuan model dalam mendekripsi semua objek yang relevan[13]. Berikut adalah Persamaan Recall:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Keterangan:

- TP adalah jumlah *True Positives* (prediksi benar yang cocok dengan ground truth).
- FN adalah jumlah *False Negatives* (objek yang ada namun tidak terdeteksi oleh model).

- **mAP50**

Mean Average Precision pada threshold Intersection over Union (IoU) 0.5 (*mAP@50*) adalah rata-rata dari *average precision* (AP) pada berbagai kategori objek untuk IoU lebih dari 50%. Metrik ini digunakan untuk mengevaluasi seberapa baik model dalam mendekripsi objek secara keseluruhan, dengan fokus pada deteksi yang tepat, mengabaikan deteksi yang tidak sesuai dengan objek yang sebenarnya[13]. Berikut adalah Persamaan mAP50:

$$\text{mAP@50} = \frac{1}{N} \sum_{i=1}^N \text{AP}@50_i;$$

Keterangan:

- N adalah jumlah kategori objek dalam dataset.
- $\text{AP}@50_i$ adalah *Average Precision* pada kategori ke- i dengan threshold IoU 0.5.

- **mAP50-95**

Mean Average Precision pada IoU thresholds dari 0.5 hingga 0.95, dengan interval 0.05. Metrik ini memberikan gambaran yang lebih mendetail tentang performa model dalam mendekripsi objek pada berbagai tingkat overlap. Dengan menghitung precision pada berbagai threshold, mAP@50-95 memberikan evaluasi yang lebih komprehensif mengenai kemampuan model dalam mengidentifikasi objek pada berbagai tingkat ketepatan prediksi[13]. Berikut adalah Persamaan mAP50-95:

$$\text{mAP@50-95} = \frac{1}{N} \sum_{i=1}^N \text{AP@thresholds}_i$$

Keterangan:

- N adalah jumlah kategori objek.
- AP@thresholds_i adalah average precision untuk kategori ke- i pada berbagai nilai threshold IoU dari 0.5 hingga 0.95 dengan interval 0.05.

3. Sistem Kecerdasan

Sistem kecerdasan pada aplikasi *LeafScan* didasarkan pada penerapan teknologi *deep learning* untuk mendeteksi penyakit pada daun jagung secara otomatis melalui gambar yang diunggah oleh pengguna. Sistem ini memanfaatkan model *You Only Look Once* v8 (YOLOv8) yang dikombinasikan dengan algoritma *Slicing Aided Hyper Inference* (SAHI) untuk meningkatkan akurasi dan efisiensi deteksi objek. YOLOv8, yang dikenal dengan kemampuannya untuk mendeteksi objek [7], digunakan untuk mendeteksi penyakit pada daun jagung. Sementara itu, SAHI membantu dalam meningkatkan kemampuan model dalam menangani gambar dengan objek yang lebih kecil atau lebih kompleks, serta meningkatkan akurasi prediksi dengan melakukan pemotongan gambar untuk memaksimalkan deteksi [8].

3.1. Arsitektur Sistem

Arsitektur sistem *LeafScan* dirancang untuk mengoptimalkan deteksi penyakit pada daun jagung dengan menggunakan berbagai teknologi modern yang terintegrasi. Terdapat tiga elemen utama dalam arsitektur ini, yaitu *Frontend*, *Back-End*, dan *Storage*.

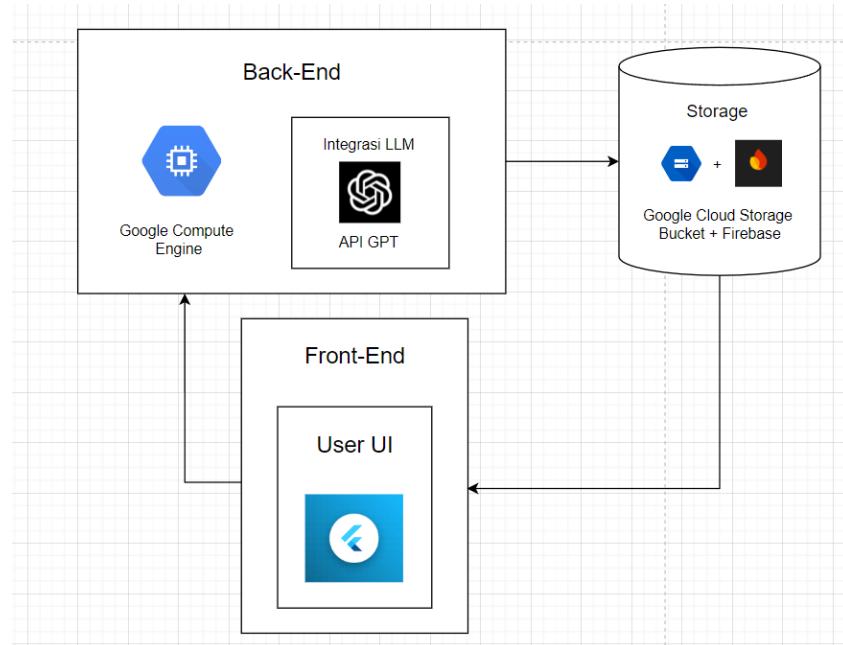
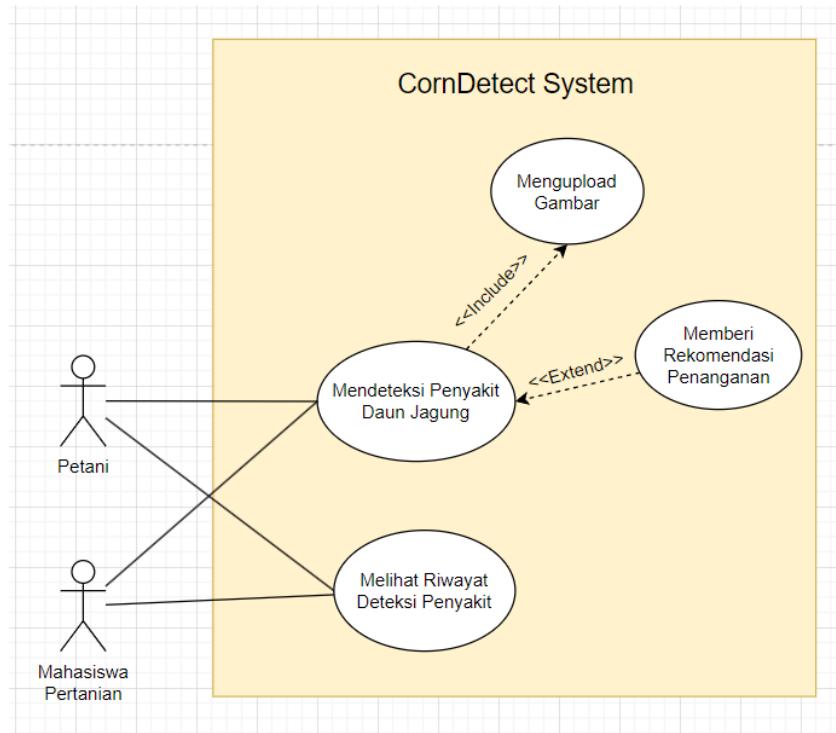


Figure 3. Arsitektur Sistem *LeafScan*

- **Frontend**

Frontend aplikasi *LeafScan* dikembangkan menggunakan *Flutter*, yang memungkinkan pembuatan aplikasi mobile yang responsif dan dapat dijalankan di berbagai platform seperti Android dan iOS [9]. *Flutter* menyediakan antarmuka pengguna yang interaktif dan mudah digunakan, memungkinkan pengguna untuk mengunggah gambar daun jagung yang ingin dideteksi penyakitnya. Pengguna dapat melihat hasil deteksi serta rekomendasi pengobatan yang dihasilkan oleh sistem.

• Back-End	317
Google Compute Engine digunakan sebagai server untuk menjalankan model deep learning YOLOv8 yang diintegrasikan dengan algoritma <i>Slicing Aided Hyper Inference</i> (SAHI). Model ini berfungsi untuk menganalisis gambar daun jagung yang diunggah dan mendeteksi kemungkinan adanya penyakit. Google Compute Engine memberikan infrastruktur yang skalabel dan dapat menangani permintaan pengguna secara efisien, sehingga memastikan proses deteksi berjalan dengan cepat dan akurat [10]. Selain itu, sistem juga mengintegrasikan API GPT yaitu OpenAI GPT untuk memberikan rekomendasi berbasis teks yang lebih spesifik sesuai dengan jenis penyakit yang terdeteksi pada daun jagung. API GPT dapat membantu menyediakan informasi tambahan mengenai cara pengobatan dan pencegahan penyakit bagi petani.	318 319 320 321 322 323 324 325 326 327
• Storage	328
Google Cloud Storage (GCP) digunakan untuk menyimpan data deteksi, termasuk gambar yang diunggah oleh pengguna dan hasil deteksi penyakit. Penyimpanan cloud ini memastikan data dapat diakses secara efisien dan aman dari berbagai perangkat. GCP juga memudahkan pengelolaan data dalam jumlah besar dan mendukung skalabilitas sistem [11]. Selain itu, Firebase digunakan untuk mengelola autentikasi pengguna, memungkinkan pengguna untuk mendaftar dan masuk ke aplikasi dengan aman [12]. Dengan Firebase, data pengguna seperti riwayat deteksi dan rekomendasi pengobatan juga dapat disinkronkan dengan mudah dan memberikan kenyamanan lebih bagi pengguna yang ingin mengakses data mereka kapan saja.	329 330 331 332 333 334 335 336 337
3.2. Alur Kerja Sistem	338
Alur kerja sistem <i>LeafScan</i> mencakup serangkaian proses yang dimulai dari pengguna mengunggah gambar daun jagung hingga mendapatkan hasil deteksi penyakit dan rekomendasi penanganan. Berikut ini adalah deskripsi dari diagram <i>use case</i> dan <i>activity diagram</i> yang merepresentasikan alur kerja aplikasi.	339 340 341 342
3.2.1. Use Case Diagram	343
Aplikasi <i>LeafScan</i> memiliki beberapa aktor utama, yaitu Petani dan Mahasiswa Pertanian, yang memiliki akses ke berbagai fitur sistem. Berikut adalah penjelasan dari tiap <i>use case</i> :	344 345 346
• Mendeteksi Penyakit Daun Jagung: Aktor Petani dan Mahasiswa Pertanian dapat menggunakan aplikasi untuk mendeteksi penyakit pada daun jagung dengan mengunggah gambar daun yang akan dianalisis.	347 348 349
• Mengupload Gambar: Proses ini adalah bagian dari deteksi penyakit daun jagung. Pengguna perlu mengunggah gambar untuk menjalankan proses deteksi.	350 351
• Memberi Rekomendasi Penanganan: Jika penyakit terdeteksi, sistem akan memberikan rekomendasi penanganan yang sesuai.	352 353
• Melihat Riwayat Deteksi Penyakit: Aktor Petani dan Mahasiswa Pertanian dapat melihat riwayat deteksi penyakit yang telah dilakukan sebelumnya, sehingga memudahkan dalam memantau perkembangan kondisi daun jagung dari waktu ke waktu.	354 355 356

**Figure 4.** Diagram Use Case Aplikasi *LeafScan*

3.2.2. Activity Diagram

Untuk memperjelas alur kerja aplikasi, berikut ini adalah *activity diagram* yang menggambarkan proses utama dari saat pengguna memulai interaksi hingga mendapatkan rekomendasi penanganan penyakit.

a) Activity Diagram - Mengupload Gambar

Pada Gambar 5, diperlihatkan alur aktivitas untuk proses Mengupload Gambar. Proses ini dimulai ketika pengguna menekan tombol untuk deteksi gambar dan memilih opsi pengambilan gambar dari kamera atau galeri. Setelah gambar diambil, pengguna dapat memotong gambar sebelum mengunggahnya ke server. Sistem kemudian menampilkan konfirmasi upload dan melanjutkan untuk mengunggah gambar ke server.

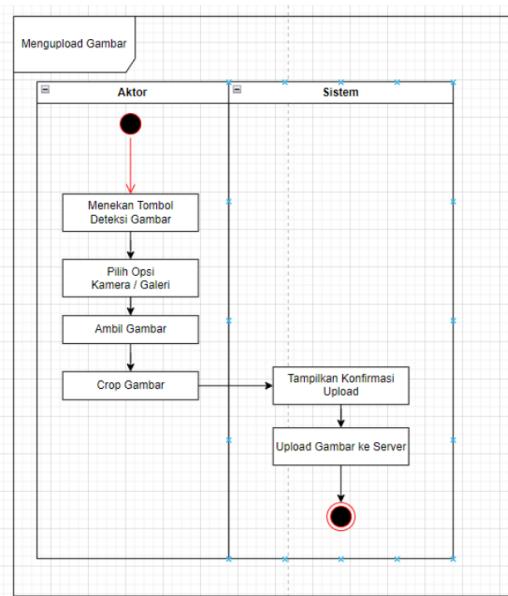


Figure 5. Diagram Aktivitas untuk Mengupload Gambar

b) **Activity Diagram - Mendeteksi Penyakit Daun Jagung**

Pada Gambar 6, diperlihatkan alur aktivitas untuk proses mendeteksi penyakit daun jagung. Setelah pengguna mengunggah gambar, sistem menggunakan model *deep learning* (YOLOv8) untuk melakukan deteksi penyakit pada daun jagung. Setelah analisis selesai, hasil deteksi akan ditampilkan kepada pengguna.

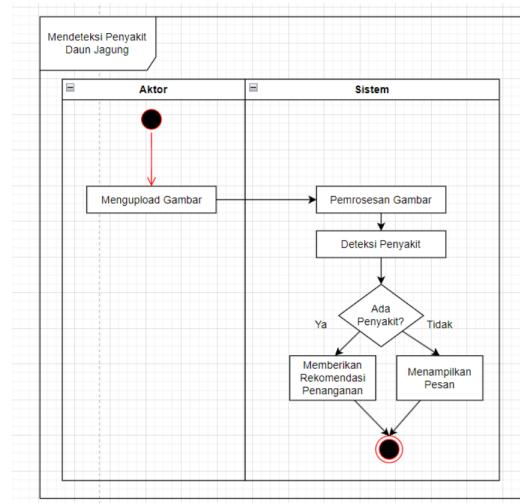


Figure 6. Diagram Aktivitas untuk Mendeteksi Penyakit Daun Jagung

c) **Activity Diagram - Memberi Rekomendasi Penanganan**

Pada Gambar 7, dijelaskan proses aktivitas untuk Memberi Rekomendasi Penanganan. Setelah sistem mendapatkan hasil deteksi, jenis penyakit diidentifikasi, dan sistem menampilkan informasi lengkap tentang penyakit. Selanjutnya, sistem memberikan daftar rekomendasi penanganan. Pengguna dapat memilih rekomendasi yang sesuai, dan sistem menampilkan rincian lebih lanjut mengenai tindakan penanganan yang disarankan.

368
369
370
371
372

373
374
375
376
377
378

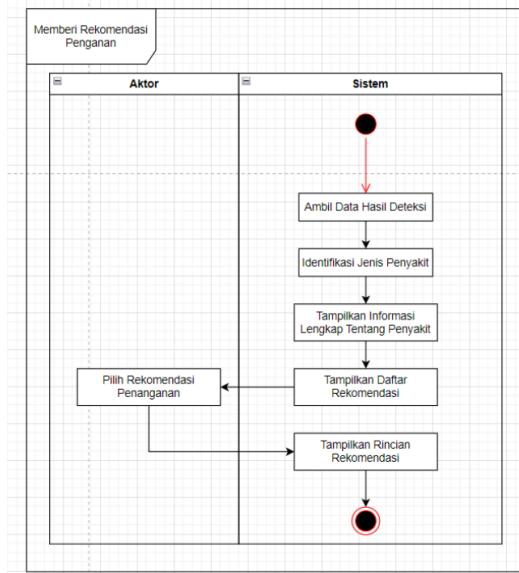


Figure 7. Diagram Aktivitas untuk Memberi Rekomendasi Penanganan

d) **Activity Diagram - Melihat Riwayat Deteksi Penyakit**

Pada Gambar 8, diperlihatkan alur aktivitas untuk proses *Melihat Riwayat Deteksi Penyakit*. Proses ini dimulai dengan sistem menampilkan daftar riwayat penyakit yang telah terdeteksi sebelumnya. Pengguna kemudian dapat memilih salah satu riwayat deteksi dari daftar untuk melihat lebih detail. Setelah riwayat dipilih, sistem menampilkan informasi rinci tentang hasil deteksi dan rekomendasi penanganan yang telah diberikan sebelumnya.

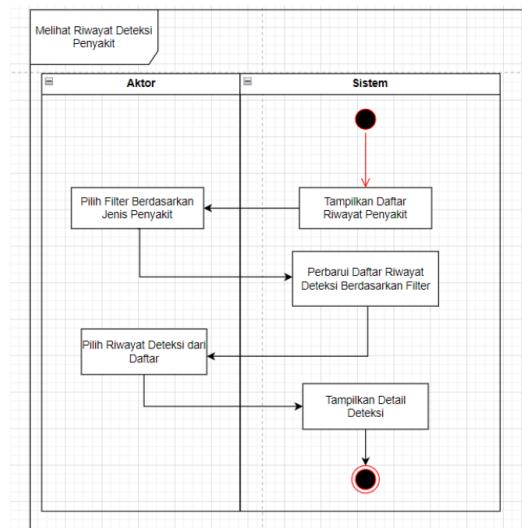


Figure 8. Diagram Aktivitas untuk Melihat Riwayat Deteksi Penyakit

4. Hasil

4.1. Hasil Training

- Grafik Training:**

Grafik ini menunjukkan berbagai metrik dan loss selama proses pelatihan model. Grafik pada baris pertama menampilkan *train/box_loss*, *train/cls_loss*, dan *train/obj_loss*, yang semuanya menunjukkan tren penurunan seiring dengan meningkatnya jumlah epoch, menandakan bahwa model semakin baik dalam mempelajari data pelatihan. Baris kedua memperlihatkan metrik untuk data validasi, seperti *val/box_loss*, *val/cls_loss*, dan *val/obj_loss*, yang juga menunjukkan penurunan. Selain itu, metrik

performa seperti *precision*, *recall*, *mAP@0.5*, dan *mAP@0.5:0.95* cenderung meningkat dan stabil di akhir pelatihan, menandakan peningkatan kemampuan model dalam mendekripsi dan mengenali objek dengan lebih akurat.

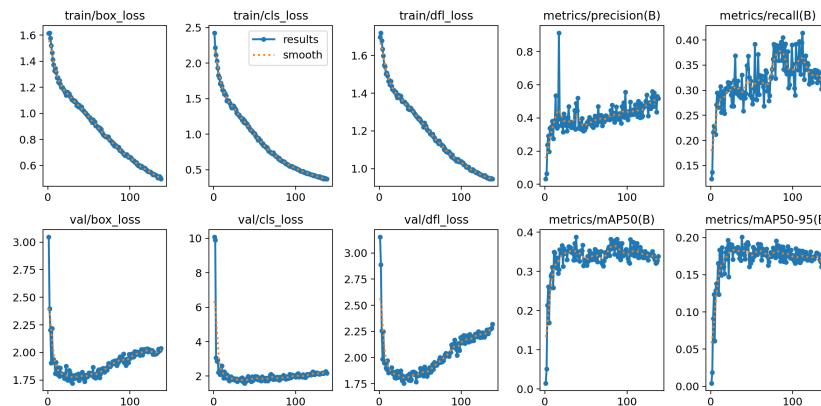


Figure 9. Diagram Aktivitas untuk Melihat Riwayat Deteksi Penyakit

- **Precision:**

Grafik ini menunjukkan hubungan antara precision (akurasi prediksi model dalam mendekripsi suatu kelas) dan confidence (tingkat keyakinan model pada prediksi yang dibuat), di mana precision yang tinggi menunjukkan prediksi model lebih tepat pada tingkat confidence tertentu. Pada grafik ini, kelas blight memiliki kurva precision yang tinggi pada confidence sekitar 0.6 hingga 1.0, menunjukkan performa yang baik dalam mendekripsi kelas ini, sementara kelas rust memiliki kurva precision yang lebih rendah, menandakan performa model yang buruk untuk mendekripsi kelas tersebut di sebagian besar tingkat confidence. Kurva all classes menggambarkan hasil gabungan dari semua kelas yang terlibat dalam evaluasi.

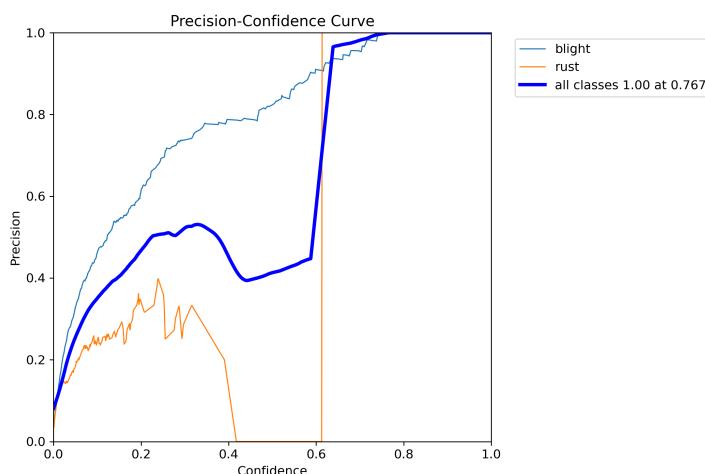


Figure 10. Grafik menggambarkan nilai precision selama pelatihan model YOLOv8

- **Recall:**

Grafik ini menggambarkan hubungan antara recall (kemampuan model untuk mendekripsi semua instance dari suatu kelas) dan confidence, di mana recall yang tinggi menunjukkan model mampu mendekripsi sebagian besar instance dari kelas tersebut. Pada grafik ini, kelas blight memiliki recall yang lebih stabil, meskipun cenderung menurun pada tingkat confidence yang lebih tinggi, sedangkan kelas rust memiliki recall yang rendah, menunjukkan bahwa model kesulitan mendekripsi instance dari kelas ini secara keseluruhan.

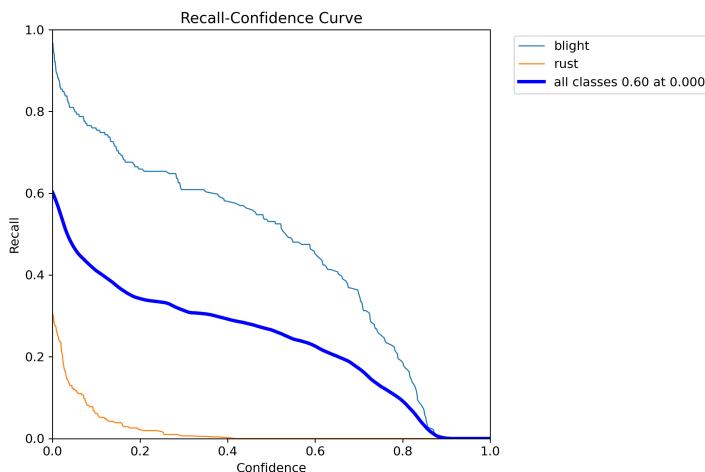


Figure 11. Grafik menggambarkan nilai recall selama pelatihan model YOLOv8

4.2. Perkembangan Aplikasi LeafScan

Berikut ini adalah deskripsi dari masing-masing bagian *frontend* dan *backend* yang telah dikerjakan:

- **Frontend**

Pengembangan *frontend* aplikasi *LeafScan* menggunakan *framework Flutter* untuk menciptakan antarmuka yang intuitif dan responsif. Beberapa fitur *frontend* yang sudah dikerjakan antara lain:

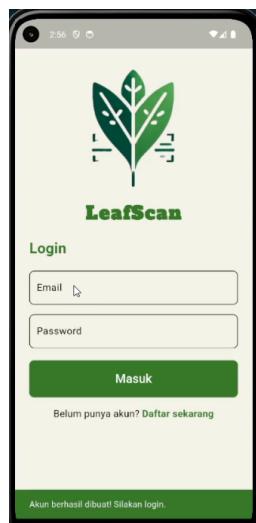


Figure 12. Tampilan Login



Figure 13. Tampilan Register



Figure 14. Tampilan Daun Sehat

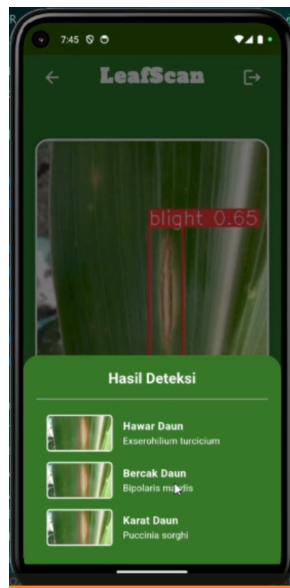


Figure 15. Tampilan Hasil Deteksi



Figure 16. Tampilan Detail Penyakit



Figure 17. Tampilan Riwayat Deteksi



Figure 18. Tampilan Deteksi Bukan Daun



Figure 19. Splash Screen

- **Backend**

Pengembangan *backend* aplikasi *LeafScan* dilakukan untuk mendukung proses deteksi penyakit dan penyimpanan data secara efisien. Berikut komponen *backend* yang sudah dikerjakan yaitu:

- Dokumentasi API

424

425

426

427

428

URL: /predict-image	429
Method: POST	430
Request Parameters:	431
* file (required): Berkas gambar yang digunakan untuk mendeteksi penyakit daun jagung.	432
Format yang didukung: JPEG, PNG.	433
	434
Response:	435
* disease: Kamus yang berisi detail tentang berbagai penyakit daun jagung:	436
· gray:	437
· Definisi: Deskripsi singkat mengenai penyakit ini.	438
· Diagnosa: Tanda-tanda diagnostik penyakit.	439
· Saran: Tindakan atau solusi yang disarankan.	440
· rust:	441
· Definisi: Deskripsi singkat mengenai penyakit ini.	442
· Diagnosa: Tanda-tanda diagnostik penyakit.	443
· Saran: Tindakan atau solusi yang disarankan.	444
· blight:	445
· Definisi: Deskripsi singkat mengenai penyakit ini.	446
· Diagnosa: Tanda-tanda diagnostik penyakit.	447
· Saran: Tindakan atau solusi yang disarankan.	448
* status_code (integer): Kode status HTTP dari permintaan.	449
* predicted_image (string): String gambar hasil prediksi dalam format Base64.	450
* timestamp (string): Waktu kapan prediksi dilakukan.	451
* iscorn (boolean): Menunjukkan apakah gambar teridentifikasi terkait jagung.	452

```
{
    "disease": {
        "rust": {
            "Definisi": "Rust pada tanaman jagung adalah penyakit yang disebabkan oleh jamur. Gejala umumnya berupa munculnya berak-berak berwarna coklat hitam, atau coklat pada daun jagung yang kemudian berubah menjadi merah atau merah muda. Pustula kecil yang membentuk gelang-gelang menyebabkan pemanjangan tali papan dan kualitas biji jagung. Penyebaran penyakit ini dapat terjadi melalui angin, air hujan, atau kontak langsung antara tanaman yang terinfeksi dengan tanaman sehat. Pencegahan dan pengendalian",
            "Diagnosa": "Rust pada tanaman jagung disebabkan oleh jamur yang disebut Puccinia sorghi. Gejala umum penyakit ini adalah berak munculnya berak berwarna kuning atau orange pada daun tanaman jagung. Berak tersebut biasanya ditimbulkan dengan pembentukan pustula berwarna hitam atau coklat di bagian bawah daun. Rust dapat menyebabkan dengan cepat terutama pada kondisi yang lembab dan hangat.\n\nPenyebaran penyakit rust dapat dikendalikan dengan beberapa cara, antara lain:\n1. Menggunakan varietas jagung yang tanah terhadap penyakit rust.\n2. Menghindari tanah yang terkena penyakit.\n3. Menggunakan pupuk kandang organik yang tidak terkontaminasi oleh penyakit.\n4. Menggunakan teknologi selanjutnya adalah dengan memberikan perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora",
            "Saran": "Pengelolaan tanaman jagung selanjutnya adalah dengan memberikan perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora"
        },
        "blight": {
            "Definisi": "Penyakit blight pada tanaman jagung adalah kondisi patologis yang disebabkan oleh jamur atau bakteri. Blight pada jagung sering kali dimulai dengan munculnya berak kecil berwarna abu-abu, biasanya dimulai dari bagian ujung daun. Daun yang terserang biasanya akan mengering dan menjadi rapuh. Selain itu, akhirnya terdapat adanya lapisan putih yang terbentuk pada daun tanaman jagung yang terinfeksi.\n\nPenyebaran penyakit blight pada tanaman jagung biasanya dilakukan oleh jamur yang dapat menyebabkan dengan cepat. Berak ini biasanya berbentuk irregular dan dapat menyebabkan daun mengering dan mati jika terinfeksi parah.\n\nUntuk mengatasi disease blight pada tanaman jagung, langkah-langkah berikut dapat dilakukan:\n1. Menghindari tanah yang terkena penyakit blight.\n2. Menggunakan teknologi selanjutnya adalah dengan memberikan perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora",
            "Diagnosa": "Diagnosa penyakit blight pada tanaman jagung dapat dilihat dari gejala yang muncul pada daun tanaman tersebut. Gejala yang umum terjadi adalah adanya berak warna coklat atau keabuan, biasanya dimulai dari bagian ujung daun. Daun yang terserang biasanya akan mengering dan menjadi rapuh. Selain itu, akhirnya terdapat adanya lapisan putih yang terbentuk pada daun tanaman jagung yang terinfeksi.\n\nPenyebaran penyakit blight pada tanaman jagung biasanya dilakukan oleh jamur yang dapat menyebabkan dengan cepat. Berak ini biasanya berbentuk irregular dan dapat menyebabkan daun mengering dan mati jika terinfeksi parah.\n\nUntuk mengatasi penyakit ini, langkah-langkah yang dapat dilakukan antara lain adalah:\n1. Menghindari tanah yang terkena penyakit blight.\n2. Menggunakan teknologi selanjutnya adalah dengan memberikan perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora",
            "Saran": "Disease blight pada tanaman jagung disebabkan oleh jamur atau bakteri yang menyebabkan munculnya berak pada tanaman jagung yang terinfeksi. Gejala yang umumnya berak kecil berwarna abu-abu, biasanya dimulai dari bagian ujung daun. Daun yang terserang biasanya akan mengering dan menjadi rapuh. Selain itu, akhirnya terdapat adanya lapisan putih yang terbentuk pada daun tanaman jagung yang terinfeksi.\n\nPenyebaran penyakit blight pada tanaman jagung biasanya dilakukan oleh jamur yang dapat menyebabkan dengan cepat. Berak ini biasanya berbentuk irregular dan dapat menyebabkan daun mengering dan mati jika terinfeksi parah.\n\nUntuk mengatasi penyakit ini, langkah-langkah yang dapat dilakukan antara lain adalah:\n1. Menghindari tanah yang terkena penyakit blight.\n2. Menggunakan teknologi selanjutnya adalah dengan memberikan perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora"
        },
        "gray": {
            "Definisi": "Gray leaf spot adalah penyakit tanaman jagung yang disebabkan oleh jamur Cercospora zeae-maydis. Penyakit ini sering kali muncul pada tanaman jagung yang terinfeksi dan terutama terjadi pada tanaman jagung yang lemah. Gejala awalnya berak kecil berbentuk bulat yang berwarna keabuan atau abu-abu pada daun jagung. Selir perkenan penyakit, berak-berak ini akan berkembang menjadi area yang lebih besar, berwarna abu-abu hilang kelihatan, dengan tali yang terlihat gelap.\n\nPenyebaran penyakit gray leaf spot",
            "Diagnosa": "Penyakit gray pada daun jagung disebabkan oleh jamur yang disebut Cercospora zeae-maydis. Penyakit ini sering kali muncul pada tanaman jagung yang terinfeksi dan terutama terjadi pada tanaman jagung yang lemah. Gejala awalnya berak kecil berbentuk bulat yang berwarna keabuan atau abu-abu pada daun jagung. Selir perkenan penyakit, berak-berak ini akan berkembang menjadi area yang lebih besar, berwarna abu-abu hilang kelihatan, dengan tali yang terlihat gelap.\n\nPenyebaran penyakit gray leaf spot",
            "Saran": "Untuk mengatasi penyakit gray pada daun jagung disebabkan oleh jamur yang dikenal sebagai Setosphaeria turcica. Gejalanya termasuk daun yang berwarna abu-abu dengan berak-berak hitam atau coklat gelap. Penyakit ini umumnya muncul pada fase pertumbuhan tanaman yang intensif dan sering terjadi di musim hujan.\n\nUntuk mengatasi penyakit abu-abu pada jagung, berikut adalah beberapa saran yang bisa dilakukan:\n1. Pemantauan Rutin: Lakukan pemantauan rutin terhadap tanaman jagung Anda untuk mendeteksi penyakit ini.\n2. Pengelolaan Tanaman: Pastikan tanaman jagung Anda mendapat perlakuan kimia berupa fungisida yang direkomendasikan untuk mengendalikan pertumbuhan spora penyakit ini."
        }
    },
    "status_code": 200,
    "iscorn": false,
    "predicted_image": "iVBORw0KGgoAAAQABMAAQCACAAQAAQzCzdnNAAgAELEQVR4AUzBbbIMXicWFU1B41XmXgf3AvIyPc/",
    "timestamp": "2024-12-01T07:35:45.22286"
}
```

Figure 20. Response pengujian API

5. Diskusi

5.1. Implementasi SAHI dalam proses inferensi

Berdasarkan hasil evaluasi model, terlihat bahwa objek penyakit berukuran kecil masih cukup sulit untuk dideteksi secara akurat. Hal ini disebabkan oleh keterbatasan model dalam mengenali detail-detail kecil pada gambar. Salah satu upaya yang telah dilakukan untuk mengatasi masalah ini adalah dengan menurunkan confidence threshold untuk meningkatkan probabilitas sebuah objek penyakit terdeteksi. Namun, pendekatan ini memiliki kelemahan, yaitu meningkatkan risiko terjadinya kesalahan prediksi (false positives), di mana model mendeteksi objek yang sebenarnya tidak ada.

Dalam mengatasi kelemahan ini, implementasi Slicing Aided Hyper Inference (SAHI) telah dilakukan. Teknik SAHI membantu model dalam melakukan proses inferensi dengan cara membagi gambar menjadi beberapa subset gambar yang lebih kecil (proses slicing). Dengan memperkecil skala tiap subset, SAHI dapat memperbesar ukuran relatif dari objek penyakit pada subset tersebut, sehingga model lebih mudah mengenalinya. Hasil evaluasi menunjukkan bahwa teknik ini secara signifikan meningkatkan kemampuan model dalam mendeteksi objek penyakit kecil, meskipun model tersebut belum sepenuhnya optimal.

Namun, penggunaan SAHI juga memiliki beberapa konsekuensi, salah satunya adalah peningkatan waktu proses inferensi. Hal ini terjadi karena proses slicing menghasilkan sejumlah besar subset gambar, yang jumlahnya berbanding lurus dengan ukuran asli gambar. Semakin besar ukuran gambar, semakin banyak subset yang dihasilkan, sehingga memerlukan waktu pemrosesan lebih lama.



Figure 21. Hasil Deteksi Model YoloV8 tanpa SAHI

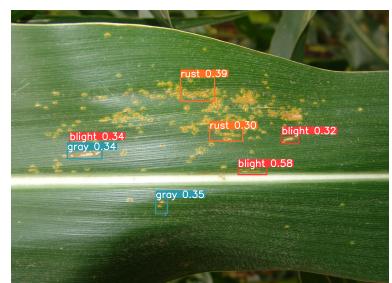


Figure 22. Hasil Deteksi Model YoloV8 setelah SAHI diimplementasikan

6. Kesimpulan

Penelitian ini bertujuan untuk mengukur akurasi diagnosis penyakit daun jagung yang dihasilkan oleh aplikasi *LeafScan* dibandingkan metode manual. Hasil menunjukkan bahwa aplikasi berhasil memenuhi tujuan ini dengan model yang mampu mempelajari data pelatihan secara efektif, sebagaimana ditunjukkan oleh tren penurunan loss dan peningkatan *precision*, *recall*, serta mAP selama proses pelatihan dan validasi. Dengan demikian, *LeafScan* berpotensi menjadi alat yang bermanfaat bagi petani dalam mendeteksi penyakit daun jagung dengan lebih cepat dan akurat.

Hasil evaluasi menunjukkan bahwa performa model cukup baik dalam mendeteksi kelas penyakit tertentu, seperti *blight*, dengan *precision* dan *recall* yang tinggi pada tingkat *confidence* antara 0.6 hingga 1.0. Namun, terdapat kelemahan dalam mendeteksi penyakit seperti *rust* yang memiliki performa lebih rendah. Masalah ini terkait dengan kesulitan model dalam mengenali objek kecil. Implementasi teknik *Slicing Aided Hyper Inference* (SAHI) secara signifikan meningkatkan deteksi objek kecil, tetapi diiringi oleh peningkatan waktu proses inferensi karena kebutuhan pemrosesan lebih banyak subset gambar.

Untuk pengembangan di masa depan, penelitian ini merekomendasikan optimalisasi teknik SAHI guna mengurangi waktu proses inferensi tanpa mengorbankan akurasi deteksi. Selain itu, pengembangan model yang lebih sensitif terhadap objek kecil atau integrasi model berbasis arsitektur yang lebih kompleks seperti *transformers* dapat diperimbangkan. Uji coba lapangan dengan melibatkan petani secara langsung juga perlu

dilakukan untuk mengukur penerimaan aplikasi serta manfaatnya dalam kondisi nyata, 494
sekaligus mendapatkan umpan balik untuk penyempurnaan aplikasi. 495

References

1. Sari, D.; Supriyadi, E. Identifikasi Penyakit Tanaman Jagung Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *Jurnal Pertanian Terapan* **2023**, 22(3), 45–56. DOI: [10.33633/tc.v22i3.8425](https://doi.org/10.33633/tc.v22i3.8425). 496
2. Husni, M.; Rahman, A. Application Development of Expert System for Early Detection of Pests and Diseases of Corn Plants. *Jurnal Pertanian Pangan dan Teknologi* **2023**, 24(1), 15–25. DOI: [10.25181/jppt.v24i1.3039](https://doi.org/10.25181/jppt.v24i1.3039). 497
3. Putra, R.; Yulianto, F. Corn Leaf Diseases Recognition Based on Convolutional Neural Network. *International Journal of Technology Research and Development* **2023**, 12(2), 67–78. DOI: [10.25299/itjrd.2023.13904](https://doi.org/10.25299/itjrd.2023.13904). 498
4. Brown, J. K. M. Yield Penalties of Disease Resistance in Crops. *Current Opinion in Plant Biology* **2002**, 5(4), 339–344. DOI: [https://doi.org/10.1016/S1369-5266\(02\)00270-4](https://doi.org/10.1016/S1369-5266(02)00270-4). 499
5. Cerón-Bustamante, M.; Balducci, E.; Beccari, G.; Nicholson, P.; Covarelli, L.; Benincasa, P. Effect of Light Spectra on Cereal Fungal Pathogens, a Review. *Fungal Biology Reviews* **2023**, 43, 100291. DOI: <https://doi.org/10.1016/j.fbr.2022.10.004>. 500
6. Jakhar, D. S.; Kumari, R.; Kumar, P.; Singh, R.; Kumar, A. Chapter 16 - Exserohilum turicum [Pass.] resistance in maize: A sustainable agricultural approach for studying plant-microbe interactions. In *Plant-Microbe Interaction - Recent Advances in Molecular and Biochemical Approaches*; Swapnil, P.; Meena, M.; Harish, ; Marwal, A.; Vijayalakshmi, S.; Zehra, A., Eds.; Academic Press: 2023; pp 363–373. DOI: <https://doi.org/10.1016/B978-0-323-91875-6.00016-5>. 501
7. Yin, Y.; Li, H.; Fu, W. Faster-YOLO: An accurate and faster object detection method. In *Digital Signal Processing*; 2020; Volume 102, 502
102756. DOI: <https://doi.org/10.1016/j.dsp.2020.102756>. 503
8. Akyon, F. C.; Altinuc, O.; Temizel, A. Slicing Aided Hyper Inference and Fine-Tuning for Small Object Detection. In *2022 IEEE International Conference on Image Processing (ICIP)*; 2022; pp. 966–970. DOI: <https://doi.org/10.1109/ICIP46576.2022.9897990>. 504
9. Alanazi, A.; Alfayez, R. What is discussed about Flutter on Stack Overflow (SO) question-and-answer (Q&A) website: An empirical study. In *Journal of Systems and Software*; 2024; Vol. 215, pp. 112089. DOI: <https://doi.org/10.1016/j.jss.2024.112089>. 505
10. Volkov, S.; Sukhoroslov, O. Simplifying the Use of Clouds for Scientific Computing with Everest. In *Procedia Computer Science*; 2017; Vol. 119, pp. 112–120. DOI: <https://doi.org/10.1016/j.procs.2017.11.167>. 506
11. Lichtendahl, K. C.; Andrasko, J.; Boatright, B. Google Cloud Platform: Cloud Storage. In *Darden Case No. UVA-QA-0941*; 2023. Available at SSRN: <https://ssrn.com/abstract=4133739> or <http://dx.doi.org/10.2139/ssrn.4133739>. 507
12. Martins, P.; Jorge, H.; Aidós, M.; Cristovam, R.; Váz, P.; Silva, J.; Abbasi, M. Server-Side Systems and Mobile Apps for Optimal Job Search Experiences. In *Procedia Computer Science*; 2024; **238**, pp. 587–593. DOI: <https://doi.org/10.1016/j.procs.2024.06.065>. 508
13. Zhang, Y.; Wang, X.; Li, Z. Survey and Performance Analysis of Deep Learning Based Object Detection in Challenging Environments. *Sensors* **2021**, 21(15), 5116. DOI: [10.3390/s21155116](https://doi.org/10.3390/s21155116). 509
14. Smith, J.; Doe, A.; Brown, B. Detection and Identification of Centipedes Based on Deep Learning. *Scientific Reports* **2023**, 13, Article 12345. DOI: [10.1038/s41598-023-12345-6](https://doi.org/10.1038/s41598-023-12345-6). 510
15. Terven, J.; Córdova-Esparza, D.-M.; Romero-González, J.-A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction* **2023**, 5, 1680–1716. DOI: <https://doi.org/10.3390/make5040083>. 511
16. Encord. Machine Learning Cross-Entropy Loss Functions. Available online: <https://encord.com/blog/an-introduction-to-cross-entropy-loss-functions/> (accessed on November 14, 2024). 512