ISSN: 2477-5126
e-ISSN:2548-9356

Deteksi Penyakit Tanaman Cabai Menggunakan Algoritma YOLOv5 Dengan Variasi Pembagian Data

Laurenza Setiana Riva[1*)], Jayanta[2]
[1,2]Jurusan Informatika, Fakultas Ilmu Komputer, Universitas Pembangunan Nasional Veteran Jakarta
[1,2]Jl. R.S Fatmawati No. 1, Cilandak, Jakarta Selatan 12450, Indonesia
email: [1]laurenzariva@gmail.com, [2]jayanta@upnvj.ac.id

Abstract* – *Rapid technological developments have resulted in various innovative techniques that help humans, including object detection which functions to identify each element in an image. Object detection is often used to overcome problems that occur because of its ability to identify each element in the image. One of the problems that is often encountered is a decrease in agricultural income due to disease in chili plants. The maintenance of chili plants has various obstacles including the impact of weather which causes the development of diseases and pests so that chili production has decreased. By implementing the object detection, farmers can easily identify diseases that attack chili plants through pictures so that chili disease can be treated more quickly. This study uses the YOLOv5 algorithm to test the performance of the model in identifying diseases in chili plants. Pictures were taken using a cellphone camera with dimensions of 3472x3472 pixels. The amount of image data used is 430 data. Image data is divided into 3 parts, namely train data, validation data, and test data. To get the best model, this study also conducted three experiments with different distribution of data. Experiment 1 with a division of 70:20:10, experiment 2 with a division of 75:15:10, and experiment 3 with a division of 80:10:10. From the experiments carried out, the best results were obtained, namely in experiment 3 with an average value obtained in the test of 0.947 with a translation of the precision, recall, and mAP values, namely 0.946, 0.936, and 0.959 respectively.

***Abstrak* – Perkembangan teknologi yang cepat telah menghasilkan berbagai teknik inovatif yang membantu manusia, termasuk *object detection*. *Object detection* sering digunakan untuk mengatasi permasalahan yang terjadi karena kemampuannya dalam mengidentifikasi setiap elemen dalam gambar. Salah satu permasalahan yang kerap ditemui yaitu penurunan pendapatan hasil pertanian akibat penyakit pada tanaman cabai. Pemeliharaan tanaman cabai memiliki berbagai kendala diantaranya akibat dampak cuaca yang menyebabkan berkembangnya penyakit dan hama sehingga produksi cabai mengalami penurunan. Dengan penerapan *object detection*, memudahkan petani mengidentifikasi penyakit yang menyerang tanaman cabai melalui gambar sehingga penanganan penyakit lebih cepat. Penelitian ini menggunakan algoritma YOLOv5 untuk menguji kinerja model dalam mengidentifikasi penyakit tanaman cabai. Pengambilan gambar dilakukan menggunakan kamera ponsel bermensi 3472x3472 piksel. Jumlah data gambar yang digunakan sebanyak 430 data. Data gambar dibagi menjadi 3 bagian yaitu data *train*, data *valid*, dan data *test*. Untuk mendapatkan model terbaik maka penelitian ini juga melakukan tiga percobaan dengan pembagian data yang berbeda. Percobaan 1 dengan pembagian 70:20:10, percobaan 2 dengan pembagian 75:15:10, dan percobaan 3 dengan pembagian 80:10:10. Dari percobaan yang dilakukan didapatkan hasil terbaik yaitu pada percobaan 3 dengan nilai rata-rata yang diperoleh pada pengujian sebesar sebesar 0.947 dengan penjabaran nilai *precision*, *recall*, dan mAP yaitu masing-masingnya sebesar 0.946, 0.936, dan 0.959.**

***Kata Kunci* – *Object Detection*, Penyakit Cabai, YOLOv5**

I. PENDAHULUAN

Banyak dari penduduk di Indonesia yang gemar makanan pedas. Mereka sering menggunakan cabai sebagai bumbu untuk berbagai hidangan seperti Mie Aceh, Balado, Bubur Pedas Sambas, Ayam Taliwang, Ayam Betutu, Seblak, dan lain-lain. Cabai rawit menjadi varietas cabai yang sangat diminati. Badan Pusat Statistik (BPS) mencatat bahwa produksi cabai rawit di Indonesia meningkat rata-rata 13,6% per tahun selama lima tahun terakhir (2016-2020). Konsumen cabai rawit terbesar adalah sektor rumah tangga yang mencapai 76,1% dari total konsumsi cabai rawit nasional pada tahun 2020 sebanyak 479,03 ton.

Pembudidayaan terhadap tanaman cabai yang dilakukan petani cabai cukup berat. Dari informasi yang didapatkan pada salah satu perkebunan tanaman cabai yaitu kebun berseri menunjukkan penurunan produksi sebesar 25% setiap tahunnya. Hal tersebut tidak lepas dari keberadaan penyakit dan hama pada tanaman cabai. Salah satu faktor yang mendukung adalah faktor cuaca. Pada saat musim kemarau banyak dari hama berkembang dengan pesat sedangkan pada saat musim hujan kondisi tanaman memiliki kelembapan yang tinggi sehingga sangat mendukung keberlangsungan hidup penyakit patogen pada tanaman.

Kemunculan penyakit dan hama pada tanaman cabai membuat petani harus waspada. Petani perlu tindakan yang akurat untuk pertumbuhan cabai yang sehat. Namun, seringkali petani mengalami kesulitan membedakan penyakit sehingga mengakibatkan penanganannya menjadi tidak optimal. Dalam hal ini diperlukan solusi bagi para petani sehingga mereka dapat mengidentifikasi dan mengatasi penyakit serta mencegahnya sehingga produksi cabai tetap stabil.

Dengan perkembangan teknologi yang semakin canggih diharapkan mampu mengatasi permasalahan yang terjadi pada petani cabai. Beberapa penelitian sudah membahas mengenai pendeteksian penyakit cabai menggunakan algoritma machine learning sederhana ataupun kompleks seperti CNN (Convolution Neural Network) seperti pada penelitian berjudul "Deteksi Penyakit Cabai Menggunakan Metode Convolution Neural Network" dan "Deteksi Penyakit pada Daun Cabai berdasarkan Fitur HSV dan GLCM menggunakan Algoritma C4.5 berbasis Raspberry Pi". Pada penelitian ini berfokus pada klasifikasi yang mana mengetahui satu jenis penyakit pada satu

gambar. Namun untuk kasus nyata seringkali pada satu gambar terekam beberapa penyakit didalamnya. Berdasarkan penelitian berjudul "*Disease Detection of Occlusion and Overlapping Tomato Leaves Based on Deep Learning*", salah satu teknologi yang dapat digunakan yaitu *object detection* yang menggunakan YOLOv3tiny. O*bject detection* bekerja dengan cara yang sama dengan bagaimana manusia melihat yaitu dari sebuah gambar dapat mengenali obyek dan nama dari obyek tersebut. obyek yang dikenali pun tidak hanya satu tapi juga dapat mengenali beberapa obyek. Pada algoritma YOLO (You Only Look Once), seluruh proses deteksi obyek dilakukan dalam satu feedforward pass melalui jaringan neural. Ini berarti bahwa jaringan YOLO langsung menghasilkan prediksi obyek dan bounding box dalam satu kali pengolahan, tanpa memerlukan tahapan-tahapan terpisah. Hal inilah yang menyebabkan mengapa YOLO disebut "You Only Look Once," karena ia hanya melihat gambar sekali untuk menghasilkan prediksi obyek sementara algoritma lain (seperti R-CNN dan SSD) memerlukan beberapa tahap atau lebih banyak komputasi. Algoritma YOLO pun semakin berkembang untuk lebih mengoptimalkan kinerja pendeteksian obyek. Penelitian ini menggunakan metode YOLOv5 dan data gambar untuk melatih komputer mengenali jenis penyakit pada tanaman cabai. Diharapkan teknologi ini membantu petani dalam mengatasi penurunan produksi cabai akibat penyakit.

*) **penulis korespondensi**: Laurenza Setiana Riva
Email: laurenzariva@gmail.com

II. PENELITIAN YANG TERKAIT

Srivastava et al, dalam penelitiannya berjudul "Comparative analysis of deep learning image detection algorithms" membahas beberapa algoritma deep learning diantaranya yaitu Single Shot Detection (SSD), Faster Region based Convolutional Neural Networks (Faster R-CNN), dan You Only Look Once (YOLO). Ketiga alogitma ini dibandingkan untuk menemukan algoritma mana yang tercepat dan paling efisien. Hasil dari penelitian ini menunjukkan bahwa YOLOv3 mengungguli SSD dan Faster R-CNN. [1]

Aini et al, dalam penelitiannya berjudul 'Deteksi dan Pengenalan Objek dengan Model Machine Learning: Model Yolo' yang dijelaskan performa tiap versi YOLO. Penelitian ini membahas perkembangan algoritma dalam memperbaiki versi sebelumnya. Penelitian ini menjelaskan mengenai peningkatan performa dari versi ke versi yang lebih tinggi mulai dari versi YOLO v1, YOLO 9000, YOLO v3, sampai YOLO v4. Namun untuk YOLOv5 belum dibahas pada penelitian ini karena masih belum memiliki sumber karya ilmiah yang memadai sejak saat penelitian tersebut dipublikasikan [2].

Sahla Muhammed Ali, dalam penelitiannya paper berjudul 'Comparative Analysis of YOLOv3, YOLOv4 and YOLOv5 for Sign Language Detection' membandingkan tiga algoritma YOLO. Hasil nilai F1-score pada masing-masing algoritma YOLOv3, YOLOv4, dan YOLOv5 yaitu 0.53, 0.63, dan 0.655. Untuk nilai mAP pada masing-masing algoritma YOLOv3, YOLOv4, dan YOLOv5 yaitu 0.46, 0.607, dan 0.633. Perbadingan algoritma YOLOv3, YOLOv4 dan YOLOv5 didapat bahwa YOLOv5 memiliki akurasi yang lebih baik dibanding versi sebelumnya [3].

Wang et al, dalam penelitiannya berjudul 'Disease Detection of Occlusion and Overlapping Tomato Leaves Based on Deep Learning' menggunakan algoritma YOLOv3tiny. Dengan YOLOv3tiny menghasilkan nilai mAP masing-masing dalam keadaan dalam pemisahan sebesar 98.3%, pada oklusi puing-puing 92.1%, dan tumpang tindih daun 90.2% [4].

III. TINJAUAN PUSTAKA

A. Praproses Data

Praproses merupakan tahap yang dilakukan sebelum data digunakan untuk membangun model. Tujuan dari tahap ini adalah untuk meningkatkan mutu citra sehingga kualitasnya ditingkatkan. Citra yang sudah ditingkatkan kualitasnya mempermudah pengolahan citra lebih lanjut [5]. Tahap ini diperlukan karena data yang berasal dari keadaan nyata seringkali memiliki kualitas yang rendah, sehingga perlu untuk memastikan bahwa citra dapat digunakan untuk mengembangkan model yang memiliki tingkat akurasi tinggi. Proses ini seringkali memerlukan waktu hingga 70% dari total waktu pengolahan data hingga mencapai hasil yang diharapkan [6].

B. Pembagian Data

Pada pembagian data dibedakan menjadi tiga kategori yaitu data train, data valid, dan data test. Data train dan valid digunakan untuk melatih model, sementara data test digunakan untuk mengevaluasi kinerja model setelah pelatihan selesai [7]. Data train, digunakan untuk membentuk model. Komputer perlu diajari untuk mengolah data mentah menjadi model [8]. Data valid, digunakan untuk mengukur kualitas model yang sedang dalam proses pelatihan [9]. Distribusi data valid harus mencerminkan distribusi data pengujian. Data test merupakan data yang belum pernah dilihat model sebelumnya yang mencerminkan gambaran data di dunia nyata dan digunakan untuk menguji akurasi hasil akhir dari model yang sudah dibuat [7].

C. Precision dan Recall

Precision berkaitan dengan sejauh mana ketepatan informasi yang diberikan kepada pengguna sesuai dengan hasil prediksi sistem. Sementara recall berkaitan dengan sejauh mana keberhasilan sistem dalam mengidentifikasi dan mengambil kembali informasi yang ada [10].

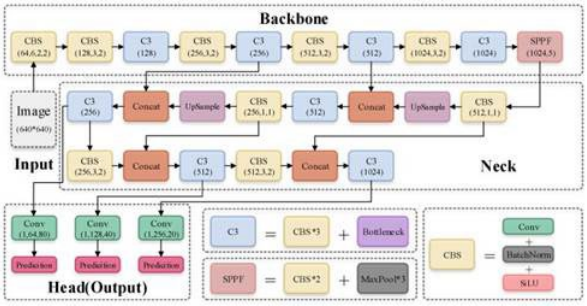
D. YOLOv5

YOLO (*You Only Look Once*) adalah suatu algoritma yang dirancang untuk mendeteksi obyek secara *real-time*. Pendekatan deteksi ini mengandalkan *localizer*, di mana model diterapkan pada berbagai lokasi dan skala pada citra. Area yang terdeteksi pada citra yang memiliki skor tertinggi dianggap sebagai hasil deteksi. Arsitektur YOLO mengadopsi *Convolutional Neural Network* (CNN) dalam *Deep Learning* yang digunakan dalam mendeteksi kebedaradaan obyek pada gambar. CNN pada YOLO hanya menggunakan lapisan konvolusi dan *pooling*. Jumlah lapisan konvolusi pada akhirnya sama dengan jumlah kelas dan jumlah kotak prediksi yang diinginkan [11].

Cara kerja dari algoritma YOLO yaitu: pertama, gambar input dibagi menjadi *grid* S×S. Jika pusat suatu obyek terletak dalam salah satu sel *grid*, sel *grid* tersebut bertanggung jawab untuk mendeteksi obyek tersebut. Kemudian, setiap sel *grid* memprediksi beberapa kotak pembatas beserta nilai *confidence* dari setiap kotak pembatas tersebut. Setiap prediksi kotak pembatas menghasilkan 5 nilai, dengan 4 nilai pertama menunjukkan letak dan ukuran kotak pembatas (x, y, w, h), dan nilai kelima adalah nilai *confidence*. Koordinat (x, y) merepresentasikan pusat kotak pembatas relatif terhadap batas sel *grid*, sedangkan nilai w dan h mengindikasikan dimensi panjang dan lebar pusat kotak relatif terhadap gambar. Sel *grid* yang mengandung obyek memprediksi probabilitas kelas yang terdeteksi di dalamnya, tanpa memperhatikan jumlah kotak pembatas [12].

YOLOv5 pertama kali diperkenalkan oleh pendiri Ultralytics Glenn Jocher. YOLOv5 menggunakan bahasa python dengan *framework* PyTorch yang mana berbeda dari pendahulunya yang menggunakan darknet dan bahasa C. Salah satu keunggulan PyTorch yaitu memiliki komunitas yang besar karena kesederhanaan dan kemudahan implementasinya. Ini juga memudahkan persiapan dan integrasi dengan perangkat IoT (*Internet of Things*) di masa depan [13].

Selain didukung oleh *framework* yang lebih sederhana untuk implementasi, YOLOv5 juga memiliki performa yang sangat cepat. Hal ini terbukti saat diuji pada Tesla P100, dengan waktu inferensi sekitar 0.007 detik per gambar atau sekitar 140 FPS (*frame* per detik) pada *default batch*. Di sisi lain, YOLOv4 setelah dikonversi ke PyTorch hanya mencapai 50 FPS. Ukuran bobot yang dihasilkan oleh YOLOv5 juga lebih kecil daripada YOLOv4. File bobot dari YOLOv5 memiliki ukuran 27 megabita, sementara YOLOv4 (dengan arsitektur Darknet) memiliki ukuran 244 megabita. Faktanya, bobot dari YOLOv5 memiliki ukuran yang 88 persen lebih kecil daripada bobot YOLOv4 [14]. Ditampilkan arsitektur YOLOv5 pada Gbr. 1 dibawah:



Gbr. 1 Arsitektur YOLOv5

Pada arsitektur YOLOv5 terdapat 3 bagian utama yaitu:

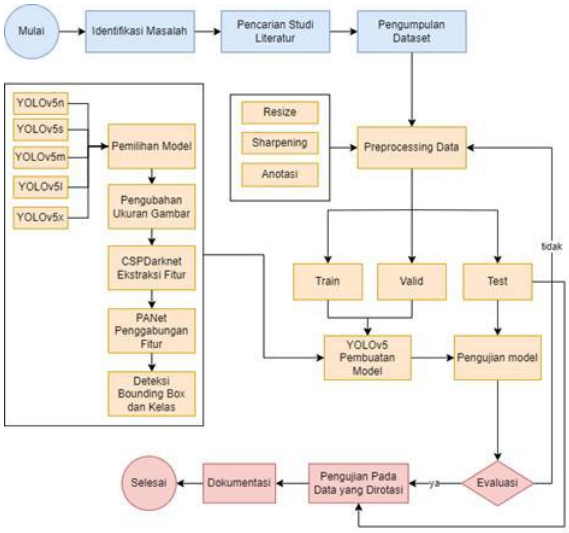
- *Backbone*
  Backbone yang digunakan YOLOv5 adalah CSPDarknet53. *Backbone* bertindak sebagai jaringan saraf konvolusional yang berfungsi untuk mengeskstrak fitur-fitur dalam gambar. CSPDarknet53 ini berperan dalam memisahkan dan menggabungkan berulang kali informasi gradien, serta menggabungkan perubahan gradien ke dalam peta fitur. Ini berdampak pada peningkatan akurasi serta efisiensi model dan juga mengurangi ukuran model dengan mengurangi jumlah parameter [15].
- *Neck*
  Neck berfungsi untuk menggabungkan dan mengolah fitur-fitur yang telah diekstrak dari gambar, yang nantinya diteruskan ke lapisan prediksi [16]. Semakin kompleks suatu jaringan maka kemungkinan kehilangan informasi semakin besar. Oleh karena itu digunakan FPN (*Feature Pyramid Network*) untuk mendeteksi obyek yang lebih kecil. Salah satu varian desain fitur FPN yang diadopsi oleh YOLOv5 adalah PANet (*Path Aggregation Network*), yang bertujuan untuk meningkatkan informasi lokal di lapisan yang lebih tinggi [17].
- *Head*
  Head berfungsi untuk memprediksi kotak pembatas beserta kelas dari obyek yang terdeteksi [16]. YOLOv5 menggunakan struktur *head* yang sama dengan versi sebelumnya (YOLOv3 dan YOLOv4). Hal ini menghasilkan tiga peta fitur keluaran yang berbeda untuk mencapai kemampuan prediksi berbagai skala. Pendekatan ini memungkinkan untuk meningkatkan prediksi obyek dari yang kecil hingga yang besar dengan efisiensi yang lebih baik [17].

Proses dalam arsitektur YOLOv5 dimulai dengan memasukkan citra ke dalam komponen *backbone*, yaitu CSPDarknet53, untuk mengekstrak fitur dari gambar. Setiap fitur yang telah diekstrak kemudian digabungkan melalui PANet sebelum akhirnya dialirkan ke komponen *head*. Bagian *head* bertugas untuk mendeteksi obyek-obyek yang ada pada gambar [15]. Terdapat tiga tahap deteksi dalam YOLOv5 yang berbeda untuk memastikan deteksi obyek dari skala kecil hingga besar dalam gambar. YOLOv5 menggunakan tiga skala yang berbeda dalam setiap *grid*, yaitu 8, 16, dan 32. Misalnya, dalam gambar ukuran 640x640 piksel, dihasilkan *grid* berukuran 80x80 untuk mendeteksi obyek kecil, 40x40 untuk obyek sedang, dan 20x20 untuk obyek besar. Tiap skala ini memiliki *grid* yang menghasilkan 3 kotak *anchor* dengan ukuran berbeda. Dengan begitu, total kotak pembatas yang dihasilkan adalah $((80 \times 80) + (40 \times 40) + (20 \times 20)) \times 3 = 25200$. Hasil deteksi dari ketiga tahap ini kemudian digabungkan dan dilakukan *Non-Max Suppression* (NMS) untuk memilih satu kotak pembatas dengan *confidence* tertinggi jika terdapat beberapa kotak pembatas untuk satu obyek [18].

IV. METODE PENELITIAN

A. Tahapan Penelitian

Pada Gbr. 2 dibawah menunjukkan alur tahapan pengerjaan yang dilakukan dalam penelitian ini:



Gbr. 2 Alur Pengerjaan

Tahapan awal penelitian dimulai dengan mengidentifikasi permasalahan kemudian dilanjutkan dengan mencari studi literatur terkait permasalahan tersebut dan memilih metode yang sesuai. Setelah itu, langkah selanjutnya adalah mengumpulkan dataset yang dibutuhkan untuk penelitian. Setelah dataset terkumpul, data melalui tahap praproses sebelum dibagi menjadi tiga kategori: data *train*, *valid*, dan *test*. Data *train* dan *valid* digunakan untuk mengembangkan model yang baik, nantinya model diuji dengan data *test*. Hasil evaluasi menentukan apakah model memenuhi standar yang diinginkan. Jika tidak, tahap praproses perlu diulang untuk memperbaiki data. Setelah didapat model terbaik selanjutnya diuji pada data *test* yang dirotasi untuk menguji kemampuannya dalam mendeteksi gambar yang mengalami perubahan sudut.

B. Indentifikasi Masalah

Peneliti mengidentifikasi permasalahan yang timbul dalam untuk mendapatkan gambaran mengenai langkah-langkah penyelesaiannya. Dalam hal ini, fokus penelitian adalah mengatasi permasalahan pada tanaman cabai. Meskipun cabai memiliki permintaan yang tinggi di pasaran, produksinya menurun akibat serangan penyakit dan hama di perkebunan. Banyak petani kurang paham mengenai jenis penyakit cabai dan cara mengatasinya. Tujuan penelitian ini adalah menciptakan model untuk mengenali penyakit cabai dan memberikan informasi kepada petani mengenai jenis penyakit yang menyerang tanaman cabai beserta solusinya. Diharapkan penelitian ini dapat mengurangi serta mencegah permasalahan penyakit cabai serta meningkatkan produksi cabai.

C. Pencarian Studi Literatur

Peneliti melakukan pencarian literatur yang terkait dengan topik penelitian yang sedang diteliti. Peneliti juga memilih algoritma yang dianggap paling cocok untuk mengatasi masalah yang sedang dihadapi, dengan mengacu pada literatur yang telah ditemukan. Sumber referensi yang mendukung penelitian ini berasal dari beragam sumber dan *platform*, termasuk jurnal, makalah, buku, situs web, dan lainnya, baik dalam dan luar negeri.

D. Pengumpulan Data

Pengumpulan data gambar dilakukan di perkebunan cabai. Penelitian ini memakai beberapa jenis dari penyakit cabai diantaranya yaitu bercak daun, hawar daun, antraknosa, dan kutu kebul. Proses pengambilan gambar sendiri menggunakan kamera *ponsel* Redmi Note 8 Pro dengan resolusi gambar sebesar 3472x3472 piksel. Pengambilan gambar dilakukan pada siang hari. Peneliti mengambil gambar dari sudut pandang tertentu dan memilah gambar mana yang menunjukkan adanya penyakit dan cukup jelas. Data gambar penyakit tanaman cabai yang digunakan sebanyak 430 gambar.

E. Praproses Data

Data yang sudah dikumpulkan dan dipilah kemudian dilakukan tahapan praproses terlebih dahulu. Tahap praproses data dimulai dari merubah ukuran gambar atau *resize* gambar dari yang semula berukuran 3472x3472 piksel menjadi gambar berukuran 640x640 piksel. Hasil gambar setelah melalui proses *resize* menjadi *blur* sehingga dilakukan proses *sharpening* atau penajaman gambar.

Tahap *sharpening* merupakan langkah untuk meningkatkan ketajaman citra, yang fokus pada meningkatkan kejelasan tepi obyek dan mengurangi elemen citra yang halus. Metode yang digunakan untuk melakukan *sharpening* yaitu filter [[0, -1, 0], [-1, 5, -1], [0, -1, 0]] yang dikenal sebagai kernel *sharpening* atau *high-pass* filter. Metode ini memiliki tujuan untuk meningkatkan kejelasan atau ketajaman gambar dengan meningkatkan perbedaan kontras di sekitar batas obyek.

Setelah data yang didapatkan sudah cukup bagus kemudian data gambar siap untuk diberikan label atau yang sering disebut dengan anotasi gambar. Tahap anotasi ini adalah proses memberikan kotak *ground truth* pada setiap obyek yang muncul dalam gambar untuk menetapkan posisi obyek dengan tepat sesuai kelasnya. Dalam tahap ini, peneliti memberikan label yang sesuai dengan jenis penyakit pada setiap gambar. Dalam pemberian label, peneliti berdiskusi dengan pengelola perkebunan cabai sehingga label yang diberikan sesuai dengan penyakitnya. Proses anotasi ini dilakukan melalui aplikasi yang berbasis bahasa pemrograman python.

F. Pembagian Data

Tahap ini, peneliti mengelompokkan data menjadi tiga kategori: data *train*, data *valid*, dan data *test*. Penelitian ini mencoba untuk melakukan variasi terhadap pembagian dataset. Terdapat tiga percobaan untuk penelitian ini berdasarkan persentase pembagian data *train*, *valid*, dan *test*. Percobaan pertama dilakukan pembagian data sebesar 70% data *train*, 20% data *valid*, dan 10% data *test*. Percobaan kedua dilakukan pembagian data sebesar 75% data *train*, 15% data *valid*, dan 10% data *test*. Percobaan ketiga dilakukan pembagian data sebesar 80% data *train*, 10% data *valid*, dan 10% data *test*.

G. Pembuatan Model

Peneliti melakukan pelatihan model menggunakan arsitektur YOLOv5. Hasil akhir dari pelatihan menghasilkan bobot akhir yang dianggap optimal, yang nantinya digunakan untuk mendeteksi gambar yang diuji. Selama tahap ini, ditampilkan juga grafik-grafik yang tercipta selama pelatihan. Grafik-grafik tersebut diantarnya yaitu *precision*, *recall*,

mAP50, mAP50-95, *box loss*, *obj loss*, dan *cls loss*. Grafik mAP (*mean Average Precision*) menggambarkan sejauh mana ketepatan kotak prediksi dalam mengestimasi kotak pembatas (*bounding box*) yang sesuai dengan kotak acuan yang sebenarnya (*ground truth box*). Untuk grafik nilai *loss* dimana terdapat tiga grafik *loss* yaitu grafik *loss* dalam penempatan *box*, grafik *loss* dalam memprediksi kelas, dan grafik *loss* dalam memprediksi ada atau tidaknya obyek pada gambar tersebut. Untuk proses pelatihan model, peneliti menggunakan layanan Google Colab.

H. Pengujian Model dan Evaluasi

Peneliti menguji model yang telah dilatih menggunakan data baru yang sebelumnya tidak pernah digunakan atau terlibat dalam proses pembentukan model. Selanjutnya hasil dari pengujian dievaluasi untuk melihat seberapa efektif model dalam mendeteksi dengan akurat. Tujuan dari tahap ini adalah model didapatkan nilai dengan tingkat akurasi yang tinggi dan tingkat *loss* yang rendah. Evaluasi dalam penelitian ini melibatkan nilai dari *presisi*, *recall*, dan mAP dari model. Penilaian tersebut berdasarkan pada obyek-obyek yang terdeteksi dengan tingkat keyakinan (*confidence*) sebesar 0,5 atau lebih.

I. Pengujian Pada Data yang Dirotasi

Pada tahap ini, dilakukan pengujian pada data *test* yang telah dirotasi. Hal ini diperlukan untuk melihat apakah model mampu mempertahankan performanya pada data yang memiliki variasi sudut yang berbeda. Dalam pengujian ini, setiap gambar diputar dengan sudut rotasi sebesar 45°, 90°, 135°, 180°, 225°, 270°, dan 315°.

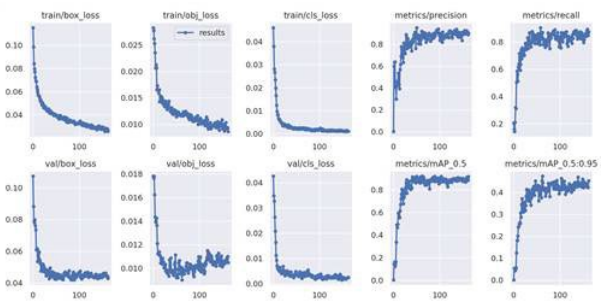
V. HASIL DAN PEMBAHASAN

Pelatihan model dilakukan dengan tiga percobaan. Setiap percobaan menghasilkan beberapa nilai diantaranya *precision*, *recall*, dan mAP50. Hasil dari setiap percobaan dapat dilihat pada Tabel I dibawah.

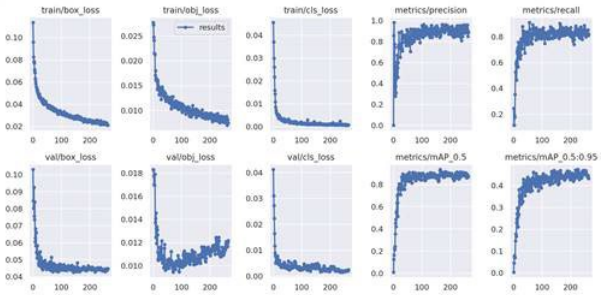
TABEL I
PERBANDINGAN KINERJA YOLOV5 PADA DATA VALID

Nilai	Percobaan 1	Percobaan 2	Percobaan 3
Precision	0.899	0.864	0.969
Recall	0.868	0.885	0.921
mAP50	0.921	0.935	0.98
Rata-rata	0.896	0.895	0.957

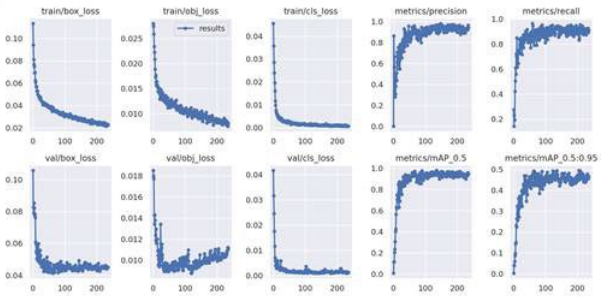
Dari Tabel I diatas dapat dilihat bahwa nilai rata-rata tertinggi pada data *valid* diraih oleh percobaan 3. Percobaan 3 mendapatkan nilai *precision*, *recall*, dan mAP50 tertinggi dengan masing-masingnya yaitu sebesar 0.969, 0.921, dan 0.98. Setiap percobaan menghasilkan 10 grafik. Grafik dapat dilihat pada gambar dibawah dengan Gbr. 3 menunjukkan grafik percobaan 1, Gbr. 4 menunjukkan grafik percobaan 2, dan Gbr. 5 menunjukkan grafik percobaan 3.



Gbr. 3 Grafik Percobaan 1.



Gbr. 4 Grafik Percobaan 2.



Gbr. 5 Grafik Percobaan 3.

Setiap percobaan dilakukan pelatihan hingga nilai yang dihasilkan tidak mengalami penambahan. Pada setiap percobaan diambil *epoch* terbaik. *Epoch* terbaik dari masing-masing percobaan yaitu percobaan 1 pada *epoch* ke 62, percobaan 2 pada *epoch* ke 164, dan percobaan 3 pada *epoch* ke 135. Hasil dari pelatihan menghasilkan file best.pt yang digunakan untuk pendeteksian obyek. Pendeteksian selanjutnya menggunakan data *test* yang merupakan data yang belum pernah dilihat model sebelumnya. Hasil deteksi obyek pada data test didapatkan hasil yang ditampilkan pada Tabel II dibawah.

TABEL III
PERBANDINGAN KINERJA YOLOV5 PADA DATA TEST

Nilai	Percobaan 1	Percobaan 2	Percobaan 3
Precision	0.915	0.976	0.946
Recall	0.876	0.903	0.936
mAP50	0.916	0.93	0.959
Rata-rata	0.902	0.936	0.947

Dari Tabel II diatas didapatkan nilai rata-rata tertinggi yang diraih oleh percobaan 3. Pada nilai *precision*, nilai tertinggi didapatkan oleh percobaan 2 dengan nilai 0.976. Pada nilai *recall* dan mAP50, nilai tertinggi didapatkan oleh percobaan 3

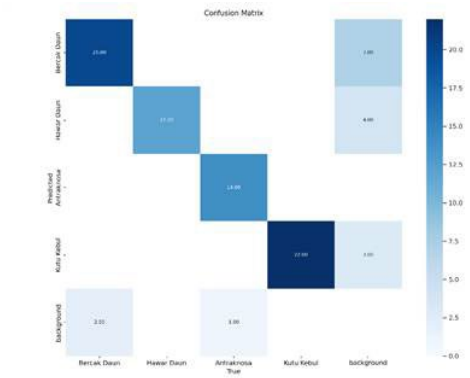
dengan nilai 0.936 dan 0.959. Hasil deteksi dapat dilihat pada gambar dibawah dengan Gbr. 6 menunjukkan hasil deteksi percobaan 1, Gbr. 7 menunjukkan hasil deteksi percobaan 2, dan Gbr. 8 menunjukkan hasil deteksi percobaan 3.

Gbr. 6 Hasil Deteksi Percobaan 1.

Gbr. 7 Hasil Deteksi Percobaan 2.

Gbr. 8 Hasil Deteksi Percobaan 3.

Dari tiga percobaan yang dilakukan didapatkan nilai rata-rata percobaan terbaik pada data *valid* dan data *test* yaitu percobaan 3 sebesar 0.957 dan 0.947. Didapatkan bahwa percobaan 3 adalah percobaan terbaik. Untuk *confusion matrix* ditampilkan pada Gbr. 9 dibawah.



Gbr. 9 Confusion Matrix.

Dari Gbr. 9 di atas, terlihat hasil deteksi yang diperoleh. Hasil prediksi yang cocok dengan kelas aslinya yaitu 20 bercak, 12 hawar, 14 antraknosa, dan 22 kutu kebul. Meskipun begitu, terdapat juga kesalahan deteksi, termasuk prediksi *background* sebagai kelas tertentu dan sebaliknya. *Background* yang terprediksi sebagai kelas memiliki rincian 7 bercak, 4 hawar, dan 3 kutu kebul. Pada kelas yang terprediksi sebagai *background*, terdapat 2 bercak dan 1 antraknosa.

Selanjutnya dilakukan perotasian pada data *test* untuk dilihat bagaimana performanya dalam mendeteksi gambar dengan kesudutan tertentu. Penelitian ini mendeteksi data test dengan sudut 45°, 90°, 135°, 180°, 225°, 270°, dan 315°. Hasil yang didapatkan dapat dilihat pada Tabel III dibawah.

TABEL IIIII
PENGUJIAN PADA DATA TEST DENGAN VARIASI SUDUT

Sudut	Precison	Recall	mAP
45°	0.802	0.751	0.794
90°	0.705	0.743	0.787
135°	0.796	0.714	0.786
180°	0.791	0.712	0.78
225°	0.724	0.707	0.783
270°	0.76	0.734	0.79
315°	0.87	0.761	0.832

Gambar pada posisi 0° digunakan sebagai acuan untuk hasil deteksi pada sudut-sudut lainnya yaitu 45°, 90°, 135°, 180°, 225°, 270°, dan 315°. Pada sudut-sudut lain terdapat penurunan nilai jika dibandingkan dengan sudut awalnya yaitu 0°. Hal tersebut dikarenakan terdapat obyek yang tidak terdeteksi walaupun pada sudut 0° terdeteksi.

VI. KESIMPULAN

Dari penelitian yang dilakukan diatas, terdapat 3 percobaan yang dilakukan dengan membedakan variasi pembagian data. Percobaan 3 mendapatkan nilai rata-rata tertinggi baik pengujian pada data *valid* maupun data *test*. Nilai rata-rata percobaan 3 yang didapatkan pada data *valid* yaitu sebesar 0.957 dengan penjabaran nilai *precision*, *recall*, dan mAP yaitu masing-masingnya sebesar 0.969, 0.921, dan 0.98. Nilai rata-rata percobaan 3 yang didapatkan pada data test yaitu sebesar 0.947 dengan penjabaran nilai *precision*, *recall*, dan mAP yaitu masing-masingnya sebesar 0.946, 0.936, dan 0.959.

Didapatkan bahwa pembagian data berpengaruh pada performa. Hal tersebut dapat disebabkan karena adanya variasi data gambar yang tidak merata. Pembagian yang tidak optimal dapat mengakibatkan evaluasi kinerja yang bias atau tidak tepat. Apabila data *valid* atau data *test* tidak mencerminkan situasi dunia nyata, metrik performa seperti *precision* dan *recall* dapat memberikan pandangan yang keliru tentang sejauh mana kinerja model yang didapat dalam skenario nyata. Pengujian lainnya yaitu mencoba melakukan perotasian gambar dimana pada skenario nyata gambar bisa diambil dari sudut pandang mana saja. Hasil dari perotasian gambar dengan data test memiliki dampak pada hasil deteksi dikarenakan adanya pergeseran yang sekiranya pada model belum memiliki varian yang sama pada saat pelatihan.

Kelebihan dari penelitian ini yaitu model yang dilatih menghasilkan performa yang bagus dalam mengenali penyakit tanaman cabai dengan waktu yang cukup cepat. Kekurangan dari penelitian ini yaitu dataset yang digunakan kurang bervariasi dalam pengambilan sudut pandang gambar. Untuk penelitian selanjutnya dapat dilakukan peningkatan variasi data gambar. Peningkatan variasi dapat dilakukan dengan cara pengambilan gambar dari berbagai sudut. Selanjutnya, dapat ditambahkan lebih banyak variasi dalam jenis penyakit pada tanaman cabai. Serta pertimbangan pencahayaan dalam pengambilan data juga diperlukan untuk memaksimalkan kinerja model.

UCAPAN TERIMA KASIH

Puji syukur peneliti ucapkan kepada Tuhan Yang Maha Esa yang telah memberikan kasih dan anugrah-Nya sehingga penulis dapat menyelesaikan penelitian ini. Peneliti mengucapkan banyak terimakasih kepada pemilik kebun cabai yang bersedia mengizinkan peneliti untuk mengambil gambar tanaman cabai sebagai dataset untuk keperluan penelitian ini khususnya kebun cabai berseri yang bersedia untuk diwawancarai terkait penelitian ini. Peneliti juga mengucapkan terimakasih kepada pihak-pihak lain yang sudah ikut andil memberikan bantuan serta dukungan dalam pengerjaan penelitian ini yang tidak bisa disebutkan satu-persatu.

DAFTAR PUSTAKA

[1] S. Srivastava, A. V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, and V. Pattabiraman, "Comparative analysis of deep learning image detection algorithms," *J Big Data*, 2021, https://doi.org/10.1186/s40537-021-00434-w.

[2] Q. Aini, N. Lutfiani, H. Kusumah, and M. S. Zahran, "Deteksi dan Pengenalan Objek Dengan Model Machine Learning: Model Yolo," *CESS (Journal of Computer Engineering, System and Science)*, vol. 6, no. 2, p. 192, 2021, doi: 10.24114/cess.v6i2.25840.

[3] Sahla Muhammed Ali, "Comparative Analysis of YOLOv3, YOLOv4 and YOLOv5 for Sign Language Detection," *IJARIIE*, vol. 7, no. 4, pp. 2393–2398, 2021.

[4] Xuewei Wang, Jun Liu, and Guoxu Liu, "Disease Detection of Occlusion and Overlapping Tomato Leaves Based on Deep Learning," *Front. Plant Sci*., vol. 12, 2021, doi:10.3389/fpls.2021.792244.

[5] F. Arnia and K. Munadi, Pengantar Teknik Pengolahan Citra dan Visi Komputer. Yogyakarta: Ombak, 2018.

[6] D. B. Lasfeto, Machine Learning Dalam Penelitian Bidang Pendidikan. 2021.

[7] R. J. Gunawan, B. Irawan, "Pengenalan Ekspresi Wajah Berbasis Convolutional Neural Network Dengan Model Arsitektur VGG16," *eProceedings of Engineering*, vol. 8, no. 5, 2021.

[8] D. Kurniawan, Pengenalan Machine Learning dengan Python. Jakarta: PT Elex Media Komputindo, 2020.

[9] GitHub, Inc., "Does validation data impact the model performance? #6023," *[2] GitHub, Inc*. [Online]. Available: https://github.com/ultralytics/yolov5/discussions/6023. [Accessed: 01-Mar-2023].

[10] C. S. Sriyano, E. B. Setiawan, "Pendeteksian Berita Hoax Menggunakan Naive Bayes Multinomial Pada Twitter Dengan Fitur Pembobotan Tf-idf," e-Proceeding Eng. Vol.8, No.2, vol. 8, no. 2, pp. 3396–3405, 2021.

[11] C. Geraldy, C. Lubis, "Pendeteksian Dan Pengenalan Jenis Mobil Menggunakan Algoritma You Only Look Once Dan Convolutional Neural Network," *Jurnal Ilmu Komputer dan Sistem Informasi*, vol. 8, no. 2, 2020.

[12] Khairunnas, E. M. Yuniarno, A. Zaini, "Pembuatan Modul Deteksi Objek Manusia Menggunakan Metode YOLO untuk Mobile Robot," JURNAL TEKNIK ITS, vol. 10, no. 1, 2021.

[13] V. Choudhari, M. Phadtare, S. Vartak, R. Pedram, "Comparison between YOLO and SSD MobileNet for Object Detection in a Surveillance Drone," *IJSREM*, vol. 5, no. 10, 2021.

[14] T. Delleji, Z. Chtourou, "An Improved YOLOv5 for Real-time Mini-UAV Detection in No Fly Zones," pp. 174-181, 2022.

[15] U. Nepal, H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs," *Sensors*, vol. 22, no. 2, 2022.

[16] M. N Bramasta, M. Anshar, I. Nurtanio, "Prototipe Sistem Pengenalan Pelat Kendaraan Otomatis Berbasis YOLO pada Mekanisme Pintu Masuk Departemen Elektro UNHAS," *Seminar Nasional Elektroteknik dan Teknologi Informasi*, 2022.

[17] A. Bochkovskiy, C. Y. Wang, H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020.

[18] B. B. Dursa and K. K. Tune, "Developing Traffic Congestion Detection Model Using Deep Learning Approach: A Case Study of Addis Ababa City Road," 2020.