Implementasi *Model Deep Learning* dalam Deteksi dan Klasifikasi Jenis Kerusakan Jalan Aspal dengan Algoritma *You Only Look Once* (YOLO) V5

SKRIPSI

Diajukan untuk Memenuhi Sebagaian Syarat Guna
Memperoleh Gelar Sarjana Program Sastra 1 (s.1)
Dalam Ilmu Teknologi Informasi

Diajukan Oleh :

Mochammad Ali Ridho Fathoni

NIM : 2008096061

PROGRAM STUDI TEKNOLOGI INFORMASI

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI WALISONGO

SEMARANG

2024

PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini:

Nama : Mochammad Ali Ridho Fathoni

NIM : 2008096061

Jurusan : Teknologi Informasi

Menyatakan bahwa skripsi yang berjudul:

IMPLEMENTASI MODEL *DEEP LEARNING* DALAM DETEKSI DAN KLASIFIKASI JENIS KERUSAKAN JALAN ASPAL DENGAN ALGORITMA *YOU ONLY LOOK ONCE* (YOLO) V5

Secara keseluruhan adalah hasil penelitian/karya saya sendiri, kecuali bagian tertentu yang dirujuk sumbernya.

Semarang, 23 Desember 2024

Pembuat Pernyataan

Mochammad Ali Ridho Fathoni

NIM. 2008096061

KEMENTRIAN AGAMA REPUBLIK INDONESIA
UNIVERSITAS ISLAM NEGERI WALISONGO
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Prof. Dr. Hamka (Kampus III) Ngaliyan Semarang 50185
Telp. (024) 7604554 Fax. (024) 7601293

PENGESAHAN

Naskah skripsi berikut ini:

Judul : Implementasi Model Deep Learning Dalam Deteksi Dan Klasifikasi Jenis Kerusakan Jalan Aspal Dengan Algoritma You Only Look Once (Yolo) V5
Nama : Mochammad Ali Ridho Fathoni
NIM : 2008096061
Jurusan : Teknologi Informasi

Telah diujikan dalam sidang tugas akhir oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo Semarang dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam Teknologi Informasi.

Semarang, 23 Desember 2024

DEWAN PENGUJI

Penguji I
Dr. Khotibul Umam, S.T., M.Kom.
NIP. 197908272011011007

Penguji II
Dr. Masy Ari Ulinuha, S.T., M.T.
NIP. 198108122011011007

Penguji III
Siti Nur`aini, M.Kom.
NIP. 198401312018012001

Penguji IV
Mokhamad Ikhil Mustofa, M.Kom
NIP. 198808072019031010

Pembimbing I
Dr. Masy Ari Ulinuha, S.T., M.T.
NIP. 198108122011011007

Pembimbing II
Adzhal Arwani Mahfudh, M.Kom.
NIP: 199107032019031006

NOTA PEMBIMBING

Semarang, 11 Desember 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi UIN Walisongo Semarang.

Assalamu'alaikum Warahmatullah Wabarakatuh.

Dengan ini memberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan:

Judul : Implementasi *Model Deep Learning* dalam Deteksi dan Klasifikasi Jenis kerusakan Jalan Aspal dengan Algoritma *You Only Look Once* (YOLO)v5

Nama : Mochammad Ali Ridho Fathoni

NIM : *2008096061*

Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Program Studi Teknologi Informasi dan Fakultas Sains dan Teknologi untuk ujian dalam ujian munaqasah di UIN Walisongo.

Wassalamualaikum Warahmatullahi Wabarakatuh.

Pembimbing I

Dr. Masy Ari Ulinuha, ST MT

NIP: 198108122011011007

NOTA PEMBIMBING

Semarang, 11 Desember 2024

Yth. Ketua Program Studi Teknologi Informasi
Fakultas Sains dan Teknologi UIN Walisongo Semarang.

Assalamu'alaikum Warahmatullah Wabarakatuh.

Dengan ini memberitahukan bahwa saya telah melakukan bimbingan, arahan dan koreksi naskah skripsi dengan:

Judul : Implementasi *Model Deep Learning* dalam Deteksi dan Klasifikasi Jenis kerusakan Jalan Aspal dengan Algoritma *You Only Look Once* (YOLO)v5

Nama : Mochammad Ali Ridho Fathoni

NIM : *2008096061*

Jurusan : Teknologi Informasi

Saya memandang bahwa naskah skripsi tersebut sudah dapat diajukan kepada Program Studi Teknologi Informasi dan Fakultas Sains dan Teknologi untuk ujian dalam ujian munaqasah di UIN Walisongo.

Wassalamualaikum Warahmatullahi Wabarakatuh.

Pembimbing II

Adzhal Arwani M., S.Kom., M.Kom

NIP: 199107032019031006

LEMBAR PERSEMBAHAN

Dengan rasa syukur yang mendalam, dengan telah diselesaikannya skripsi ini, penulis mempersembahkan kepada:

1. Keluarga besar penulis yang senantiasa memberikan dukungan dan doa kepada penulis.
2. Segenap civitas akademik UIN Walisongo Semarang, staf pengajar, karyawan, dan seluruh mahasiswa semoga selalu dalam keadaan sehat dan tetap semangat dalam beraktivitas mengisi hari-harinya di kampus tercinta UIN Walisongo Semarang.
3. Teman teman penulis yang selalu memberikan support kepada penulis dari awal hingga akhir.

MOTTO

"Jika Kamu tidak sanggup menahan lelahnya belajar maka kamu harus sanggup menahan perihnya kebodohan."

Imam Syafi'i

ABSTRAK

Kerusakan jalan aspal, seperti retak halus, retak kulit buaya, dan lubang, dapat menimbulkan resiko keselamatan bagi penggunanya. Penelitian ini bertujuan untuk mengembangkan sistem deteksi dan klasifikasi kerusakan jalan menggunakan algoritma *You Only Look Once* (YOLO)v5 berbasis *deep learning*. Dataset terdiri dari 580 gambar yang mencakup tiga jenis kerusakan utama, yaitu retak halus, retak kulit buaya, dan lubang. Dataset ini kemudian diproses dengan teknik augmentasi, dilabeli, dan dilatih menggunakan Google Colab. Model terbaik yang dihasilkan pada proses pelatihan akan diimplementasikan dalam sistem berbasis web dengan *framework* Flask. Sistem deteksi ini dirancang untuk memberikan antarmuka yang sederhana namun tetap efektif, yang memungkinkan pengguna untuk mengunggah gambar jalan dan mendapatkan hasil deteksi secara langsung. Hasil pengujian menunjukkan akurasi deteksi yang tinggi dengan *mean Average Precision* (mAP) sebesar 0,797 dan *F1-Score* 0,79. Sistem ini mampu mendeteksi kerusakan secara cepat dan akurat dengan rata-rata waktu deteksi sekitar 0.43 detik, sehingga dapat mendukung efisiensi pemeliharaan jalan serta pengembangan teknologi deteksi berbasis *deep learning*.

Kata Kunci: YOLOv5, deep learning, deteksi kerusakan jalan, klasifikasi objek, pengolahan citra.

ABSTRACT

Damage to asphalt roads, such as fine cracks, alligator cracks, and potholes, can pose safety risks to users. This study aims to develop a detection and classification system for road damage using the You Only Look Once (YOLO)v5 algorithm based on deep learning. The dataset consists of 580 images covering three main types of damage: fine cracks, alligator cracks, and potholes. The dataset was then processed using augmentation techniques, labeled, and trained using Google Colab. The best model produced during the training process will be implemented in a web-based system using the Flask framework. This detection system is designed to provide a simple yet effective interface, allowing users to upload road images and obtain detection results instantly. Testing results indicate high detection accuracy, with a mean Average Precision (mAP) of 0.797 and an F1-Score of 0.79. The system can detect damage quickly and accurately, with an average detection time of approximately 0.43 seconds, supporting road maintenance efficiency and the development of deep learning-based detection technology.

Keywords: YOLOv5, deep learning, road damage detection, object classification, image processing.

KATA PENGANTAR

Puji dan syukur selalu saya panjatkan ke hadirat Allah SWT, atas berkat dan rahmat-Nya, sehingga penulis dapat menyelesaikan penelitian ini dengan baik. Adapun judul Skripsi yang penulis ambil sebagai berikut, "Implementasi Model Deep Learning dalam Deteksi dan Klasifikasi Jenis Kerusakan Jalan Aspal dengan Algoritma You Only Look Once (YOLO)v5.". Tujuan Skripsi pada Program Sarjana (S1) Prodi Teknologi Informasi ini dibuat sebagai salah satu syarat kelulusan Program Sarjana Universitas Islam Negeri Walisongo Semarang. Sebagai bahan diambil berdasarkan hasil penelitian, observasi dan beberapa sumber literatur yang mendukung penulisan ini. Penulis menyadari bahwa tanpa bimbingan dan dorongan dari semua pihak, maka penulisan Skripsi ini tidak akan berjalan lancar. Oleh karena itu pada kesempatan ini, izinkanlah penulis menyampaikan ucapan terimakasih kepada:

1. Bapak Prof. Dr. Nizar, M.Ag, selaku Rektor Universitas Islam Negeri Walisongo Semarang.
2. Bapak Dr. H. Ismail, M.Ag, selaku Dekan Fakultas Teknologi Informasi Universitas Islam Negeri Walisongo Semarang.
3. Bapak Dr. Khotibul Umam, ST., M.Kom selaku Ketua Program Studi Teknologi Informasi Universitas Islam Negeri Walisongo Semarang dan pembimbing pertama.

4. Bapak Dr. Masy Ari Ulinuha, ST., M.T. dan Bapak Adzhal Arwani Mahfudh, M.Kom. Selaku dosen pembimbing skripsi saya yang selalu memberikan bimbingan dan bantuannya dalam pembuatan skripsi ini.
5. Orang tua tercinta yang selalu berjuang dan menemani dalam membantu penulis untuk menggapai semua mimpi, dan memberikan dukungan baik secara moril maupun materil kepada penulis.
6. Teman-teman penulis yang selalu memberikan dukungan dan orang – orang yang mungkin tidak bisa saya sebutkan satu-satu, akan tetapi tidak mengurangi rasa terima kasih dan rasa hormat saya kepada kalian.

Dalam pelaksanaan dan penyusunan Skripsi, penulis menyadari bahwa tentunya masih jauh dari kata sempurna dan banyak kekurangan. Untuk itu, penulis sangat mengharapkan adanya kritik konstruktif serta saran yang membangun, dan semoga Skripsi ini tidak hanya menjadi catatan lapuk yang termakan oleh usia tetapi juga dapat bermanfaat untuk semua pihak.

Semarang, 23 Desember 2024

Mochammad Ali Ridho Fathoni

NIM. 2008096061

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

A. Latar Belakang

Infrastruktur jalan merupakan suatu komponen penting dalam mendukung mobilitas dan konektivas dalam masyarakat. Namun seiring dengan faktor-faktor seperti cuaca yang ekstrim, lalu lintas yang padat dan pemeliharaan jalan yang kurang tepat waktu, dapat membuat jalan rentan mengalami kerusakan, adapun beberapa kerusakan pada jalan aspal seperti retak halus, retak kulit buaya, lubang dan sebagainya. Kerusakan jalan tersebut dapat menimbulkan ketidaknyamanan bagi pengguna jalan atau bahkan dapat mengakibatkan kecelakaan pada saat berkendara. Islam menekankan mengenai pentingnya menjaga keselamatan dan kemaslahatan umat. Allah berfirman dalam Q.S Al-Baqarah ayat 195:

وَاَنْفِقُوْا فِيْ سَبِيْلِ اللّٰهِ وَلَا تُلْقُوْا بِاَيْدِيْكُمْ اِلَى التَّهْلُكَةِ وَاَحْسِنُوْا ۛ اِنَّ اللّٰهَ يُحِبُّ الْمُحْسِنِيْنَ

Artinya :

"Berinfaklah di jalan Allah, janganlah jerumuskan dirimu ke dalam kebinasaan, dan berbuatbaiklah. Sesungguhnya Allah menyukai orang-orang yang berbuat baik". (QS. Al-Baqarah Ayat 195)

Pada ayat ini menegaskan betapa pentingnya mencegah hal-hal yang dapat membahayakan diri sendiri maupun orang lain, termasuk dalam konteks infrastruktur jalan. Jalan yang rusak dapat menjadi penyebab kecelakaan yang tidak hanya merugikan individu tetapi juga masyarakat secara keseluruhan. Selain itu pemeliharaan jalan yang efisien dan tepat waktu adalah kunci dalam memastikan minimnya timbul kerusakan pada jalan(Istri Lestari et al., 2022).

Langkah awal dalam melakukan pemeliharaan jalan adalah dengan melakukan identifikasi kerusakan pada suatu jalan, sehingga dapat menentukan tindakan apa yang perlu dilakukan. Adapun metode untuk mengidentifikasi kondisi kerusakan jalan dapat dilakukan secara manual ataupun otomatis. Metode manual dilakukan dengan cara menyusuri jalan, mengambil gambar, mengukur area kerusakan, menentukan tingkat kerusakan sesuai dengan jenis kerusakan, lalu menghitung dan menuliskannya dalam bentuk laporan. Metode ini tentunya akan sangat menyita waktu, tenaga dan biaya, selain itu metode ini juga rawan subjektifitas sehingga dapat mempengaruhi akurasi dalam identifikasi kerusakan. Sedangkan identifikasi secara otomatis dapat dilakukan dengan bantuan alat yang dapat mengambil citra kondisi jalan dan secara otomatis

membedakan jenis kerusakan jalan, cara ini cenderung lebih efektif dan objektif karena deteksi dilakukan secara otomatis sehingga dapat mengurangi kesalahan dalam deteksi. Pengindentifikasian dengan cara ini juga dapat dijadikan acuan dalam memberikan tindakan yang tepat untuk pemeliharaan jalan(Suryowinoto & Hamid, 2017).

Mengingat pentingnya identifikasi jenis kerusakan jalan, penelitian ini bertujuan untuk mendeteksi dan mengklasifikasikan jenis kerusakan jalan aspal. Dalam penelitian ini, deteksi dan klasifikasi jenis kerusakan jalan dilakukan dengan menggunakan algoritma *You Only Look Once* (YOLO)v5. YOLO merupakan salah satu algoritma deteksi objek dengan berbasis Deep Learning yang dikembangkan pertama kali oleh Joseph Redmon pada tahun 2015 (Sauqi, 2022).

YOLO adalah algoritma pendeteksi objek yang dikembangkan berdasarkan metode CNN (*Convulotional Neural Network*). Algoritma ini merupakan algoritma pendeteksi objek satu langkah (*one-stage detection*) pertama dengan berbasis *deep learning*. YOLO memeriksa sebuah gambar satu kali dan mampu memprediksi objek apa saja yang ada didalam gambar beserta lokasi objek tersebut. Hal ini dimungkinkan dengan mengamati operasi YOLO dan menggunakan jaringan konvolusi untuk terus

memprediksi kotak pembatas dan kelasnya (Hidayat, 2023).

Untuk mendukung implementasi sistem deteksi otomatis ini, *framework* Flask digunakan sebagai platform pengembangan aplikasi berbasis web. Flask merupakan *framework* web Python yang ringan dan fleksibel, sehingga cocok untuk membangun aplikasi yang mengintegrasikan model YOLOv5 dengan antarmuka pengguna. Dengan Flask, hasil deteksi kerusakan jalan dapat ditampilkan secara *real-time* melalui halaman website yang interaktif, yang mana memungkinkan pengguna untuk memantau kondisi jalan secara efisien.

Algoritma YOLO terbukti merupakan metode yang lebih efisien dibandingkan dengan algoritma *machine learning* lainnya, namun sampai saat ini belum banyak diterapkan dalam pengidentifikasian citra(Pramestya, 2018). Oleh karena itu penulis tertarik menggunakan algoritma YOLO untuk mendeteksi dan mengklasifikasi citra jenis kerusakan jalan, serta memanfaatkan Flask untuk membangun sistem deteksi yang mudah digunakan dan efisien.

B. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas maka rumusan masalah dari penelitian ini adalah sebagi berikut :

1. Bagaimana mendeteksi dan mengklasifikasi jenis kerusakan jalan aspal dengan menggunakan algoritma YOLOv5?
2. Bagaimana kinerja algoritma YOLOv5 dalam mendeteksi dan mengklasifikasi jenis kerusakan jalan aspal?
3. Bagaimana cara mengintegrasikan algoritma YOLOv5 dengan framework Flask untuk membangun aplikasi berbasis web yang dapat menampilkan hasil deteksi kerusakan jalan secara real-time?

C. Batasan Masalah

Berdasarkan latar belakang dan identifikasi masalah yang telah diruaikan diatas, maka penulis membatasi masalah pada proses penelitian yaitu sebagai berikut :

1. Pada penelitian ini deteksi dan klasifikasi dilakukan dengan menggunakan algoritma YOLOv5.
2. Data gambar jalan rusak yang digunakan pada penelitian diperoleh dengan pengambilan secara langsung di Kota Semarang dan pengambilan data dari internet.

3. Penelitian difokuskan pada kerusakan jalan aspal berupa retak halus (*Hair Cracking*), retak kulit buaya (Alligator Cracking), lubang (*Potholes*),
4. Format gambar yang digunakan pada penelitian adalah JPG/JPEG.
5. Hasil deteksi dan klasifikasi berupa citra gambar.

D. Tujuan Penelitian

Berdasarkan rumusan masalah di atas, didapatkan tujuan penelitian ini adalah sebagai berikut :

1. Melakukan proses deteksi dan klasifikasi jenis kerusakan jalan aspal dengan menggunakan algoritma YOLOv5
2. Menganalisa kinerja algoritma YOLOv5 dalam mendeteksi dan mengklasifikasi jenis kerusakan jalan aspal
3. Mengintegrasikan algoritma YOLOv5 dengan framework Flask untuk membangun aplikasi berbasis web yang dapat menampilkan hasil deteksi kerusakan jalan secara real-time?

E. Manfaat Penelitian

Adapun manfaat yang dapat diperoleh dari penelitian ini adalah sebagi berikut :

1. Manfaat Teoritis

Dari hasil penelitian yang dilakukan diharapkan dapat turut berperan dalam pengembangan ilmu pengetahuan dan sebagai rujukan penelitian lain dalam bidang deteksi objek, khususnya dalam pemanfaatan model deep learning dan algoritma YOLOv5.

2. Manfaat Praktis
   a. Penelitian ini dapat digunakan oleh berbagai pihak baik dari pemerintahan maupun swasta untuk mendeteksi jenis kerusakan jalan
   b. Penelitian ini diharapkan dapat membantu dalam pengembangan sistem monitoring kerusakan jalan yang lebih canggih.

BAB II

LANDASAN PUSTAKA

A. KAJIAN TEORI

1. Kerusakan Jalan

Pada bagian ini akan dipaparkan mengenai pengertian jalan dan jenis-jenis kerusakan jalan

a. Pengertian Jalan

Jalan raya adalah suatu lintasan yang bermanfaat untuk melewatkan lalu lintas dari suatu tempat ke tempat lain. Karena jalan raya sebagai sarana perhubungan, sehingga kondisi lalu lintas harus lancar, memenuhi syarat teknis dan ekonomis sesuai fungsi, volume, dan sifat-sifat lalu lintas(Suryadharma & Susanto, 1999).

b. Jenis Kerusakan Jalan

Secara garis besar kerusakan dapat dibedakan menjadi dua bagian, yaitu kerusakan struktural dan fungsional. Kerusakan struktural yaitu kerusakan yang mencakup kegagalan perkerasan atau kerusakan dari satu atau lebih komponen perkerasan. Kerusakan jenis ini mengakibatkan perkerasan tidak dapat lagi menanggung beban lalu lintas. Sedangkan kerusakan fungsional yaitu kerusakan yang

mengakibatkan keamanan dan kenyamanan pengguna jalan terganggu. Beberapa jenis kerusakan jalan diantaranya (Yudaningrum & Ikhwanudin, 2017):

1) Retak halus (*hair cracking)*

Retak halus memiliki lebar celah lebih kecil atau sama dengan 3 mm, penyebab adalah bahan perkerasan yang kurang baik, tanah dasar atau bagian perkerasan di bawah lapis permukaan kurang stabil. Retak halus ini dapat meresapkan air kedalam lapis permukaan. Untuk pemeliharaan dapat dipergunakan lapis latasir atau buras. Dalam tahap perbaikan sebaiknya dilengkapi dengan perbaikan sistem drainase. Retak rambut dapat berkembang menjadi retak kulit buaya. Gambar 2.1 adalah contoh dari retak halus.

Gambar 2. 1 Retak halus

2) Retak kulit buaya (*alligator Crack*)

Retak jenis ini memiliki pola yang menyerupai kulit buaya, dengan lebar jaring lebih besar atau setidaknya 3 mm. Retak ini terjadi akibat beban lalu lintas yang berlebihan dan berkelanjutan. Beberapa kemungkinan penyebabnya adalah bahan perkerasan yang kurang berkualitas, sehingga jalan menjadi rapuh dan aspal mengalami pelapukan. Gambar 2.2 adalah contoh dari retak kulit buaya.

Gambar 2. 2 Retak kulit buaya

3) Lubang (*Potholes*)

Lubang adalah lekukan permukaan perkerasan akibat hilangnya lapisan aus dan material lapis pondasi. Kerusakan berbentuk

lubang kecil biasanya berdiameter kurang dari 0,9 m dan berbentuk mangkuk yang dapat berhubungan atau tidak berhubungan dengan permukaan lainnya. Lubang biasanya terjadi akibat galian utilitas atau tambalan di area perkerasan yang telah ada. Gambar 2.3 adalah contoh dari kerusakan jenis lubang.

Gambar 2. 3 Lubang

2. Pengolahan Citra Digital

Pengolahan citra digital adalah proses yang menggunakan komputer untuk memproses gambar digital. Dalam proses ini, algoritma komputer digunakan untuk memperbaiki gambar, mengekstrak informasi, atau menghasilkan gambar baru dengan kualitas yang lebih baik(Andono & Sutojo, 2018).

Citra sendiri dapat didefinisikan sebagai suatu gambar bidang dua dimensi yang tersusun atas banyak piksel (bagian terkecil dari citra). Citra juga dapat disebut sebagai gambar, secara umum citra terbentuk

atas kumpulan box-box segi empat yang tersusun secara teratur sehingga memiliki jarak antar bagian yang sama pada seluruh citra. Sebagai sebuah keluaran dari sistem perekaman data, citra memiliki beberapa sifat, yaitu optik yang dapat berupa foto, kemudian analog yang dapat berupa sinyal video seperti gambar yang muncul di layar *handphone* maupun layar televisi, serta digital yang dapat disimpan pada sebuah media penyimpanan magnetik (Sandipan, 2018).

Citra dapat dibagi menjadi dua jenis, yaitu citra diam dan citra bergerak. Citra bergerak terdiri dari sekuensial citra yang ditampilkan secara beruntun, menciptakan kesan gambar yang bergerak. Kumpulan citra ini disebut sebagai "frame." Contohnya, dalam video atau film, layar lebar terdiri dari ratusan hingga ribuan frame yang membentuk keseluruhan tampilan. Citra digital dapat dibagi menjadi tiga jenis diantaranya(Sultoni et al., 2019) :

a. Citra biner (*binary*)

   Merupakan citra yang hanya memiliki 2 nilai derajat keabuan yaitu putih dan hitam. Piksel dari warna tersebut memiliki nilai 1 yang berarti warna putih dan nilai 0 yang memiliki warna hitam.

b. Citra keabuan (*grayscale*)

Yaitu citra yang pada setiap piksel memiliki satu lapisan dimana memiliki nilai intensitas dengan nilai 0 (hitam) – 255 (putih).

c. Citra warna

Citra berwarna adalah citra digital dengan informasi warna dalam tiap piksel. Ada beberapa sistem warna untuk gambar berwarna, seperti RGB, CMYK dan HSV. RGB merupakan model warna yang umum digunakan, terusun atas tiga warna dasar hijau, merah dan biru yang digabungkan untuk membentuk rangkaian warna yang lebih luas dengan kisaran nilai dari 0 hingga 255.

3. Computer Vision

Computer vision adalah bidang ilmu komputer yang berfokus pada pembuatan sistem digital yang dapat memproses, menganalisis, dan memahami data visual (gambar atau video) dengan cara yang sama seperti yang dilakukan manusia. Dalam *Computer vision*, mesin berusaha untuk mengambil informasi visual, menanganinya, dan menginterpretasikan hasilnya melalui algoritma perangkat lunak khusus. *Computer Vision* dapat dirumuskan sebagai sebuah gabungan sistem dari kamera, komputer dan *pattern*

recognition. *Computer Vision* membuat komputer atau sistem bertindak layaknya manusia dengan memiliki indra penglihatan atau juga diartikan mampu menerima informasi secara visual (Szeliski, 2022).

Kemampuan komputer dalam menerima informasi secara visual ini terdapat beberapa macam diantaranya adalah *Object Detection* (kemampuan untuk mengenali suatu objek yang terlihat pada citra dan mengetahui batas objek yang dikenali melalui proses matematis berkaitan dengan jarak antar data), *Recognition* (kemampuan menempatkan label kelas pada suatu objek), *Interprating Motion* (kemampuan menafsirkan gerakan pada suatu citra bergerak), *3D Inference* (kemampuan untuk menafsirkan suatu citra 3D dari 2D yang nampak), dan *Description* (kemampuan untuk menugaskan property pada suatu objek)(Marr, 2010).

4. *Artificial Intelligence* (AI)

Artificial Intelligence (AI) adalah simulasi kecerdasan manusia yang diterapkan ke dalam sistem komputer atau perangkat mesin lainnya. Dengan AI, perangkat tersebut dapat berpikir dan bertindak seperti manusia. Tujuan utama AI adalah untuk membuat teknologi yang mampu meniru aktivitas

kognitif manusia, termasuk belajar, penalaran, pengambilan keputusan, dan koreksi diri. *Artificial Intelligence* merupakan sebuah konsep besar yang memiliki banyak algoritma atau metode yang terbentuk melalui pola yang diberikan ke suatu sistem. Komputer belajar mengenali pola tersebut kemudian membentuk sebuah struktur yang dapat menyelesaikan sebuah permasalahan (Hidayat, 2023).

Cara kerja AI melibatkan pengolahan data yang diinput untuk pembelajaran. Data ini digunakan sebagai sumber pengetahuan bagi AI, yang kemudian mengidentifikasi pola, menganalisis hubungan antar data, dan mengambil keputusan berdasarkan apa yang telah dipelajari. Semakin banyak data yang diolah, semakin meningkat pula kemampuan AI, mirip dengan cara kerja otak manusia (Fetzer, 1990).

5. Machine Learning

Machine learning adalah subkelas dari artificial intelligence (AI) yang merupakan pembelajaran mandiri yang bergantung pada algoritma, yang memungkinkan sistem untuk belajar dari pengalamannya. Sebagai contoh, sistem dapat mempelajari pola dari jenis data yang diberikan inputnya dan kemudian menggunakan

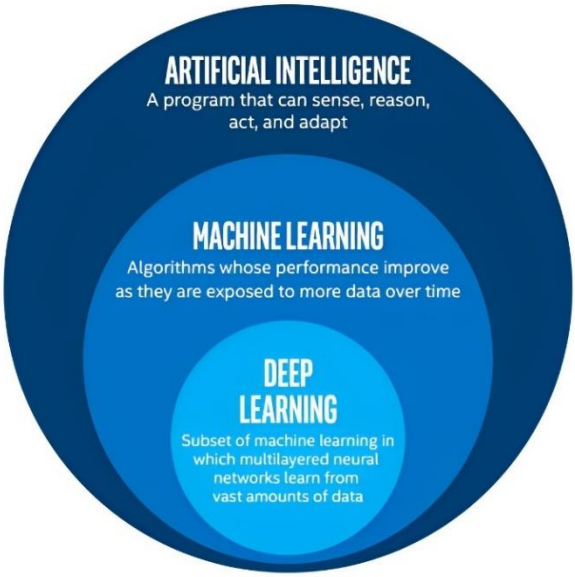
pembelajarannya untuk menghasilkan output. Dalam hal ini, sistem secara otomatis menjadi lebih pintar seiring waktu tanpa bantuan manusia karena menggunakan algoritma pembelajaran statistik yang secara otomatis belajar dan berkembang(Sharma et al., 2021).

6. Deep Learning

Deep learning merupakan bagian dari machine learning. Deep learning mengadopsi pendekatan arsitektur jaringan saraf, sehingga sering disebut sebagai *Deep Neural Networks* (DNN). Model jaringan dalam *deep learning* terdiri dari *neuron* dengan banyak parameter dan lapisan di antara *input* dan *output*. Dalam *deep learning*, sistem secara otomatis mempelajari fitur-fitur dan representasi hirarkis pada berbagai tingkat. Lapisan awal melakukan pemrosesan sederhana terhadap data input atau mempelajari fitur-fitur dasar, sedangkan outputnya diteruskan ke lapisan yang lebih tinggi untuk mempelajari fitur-fitur yang lebih kompleks(Christlein et al., 2019).

Dapat dilihat pada gambar dibawah ini, AI dapat diidentifikasi sebagai pembelajaran yang memungkinkan sebuah mesin dapat melakukan tugas yang biasanya membutuhkan kecerdasan manusia,

sebagai contoh sebuah komputer yang mampu belajar tanpa diprogram secara eksplisit langkah demi langkah dan mampu melakukan prediksi pada data yang diberikan.



Gambar 2. 4 klasifikasi Deep Learning
Sumber : *(Hamadi, 2019)*

Deep learning mempunyai beberapa tipe arsitektur diantaranya *Convolutional Neural Network* (CNN) yang digunakan untuk pengenalan 2D seperti gambar, *Recurrent* Neural Networks (RNN) digunakan untuk pengenalan suara, RBM (*Deep/Restricted*

Boltzmann Machines) dan jaringan *Long Short Term Memory* (LSTM)(Hamadi, 2019).

7. Deteksi Objek

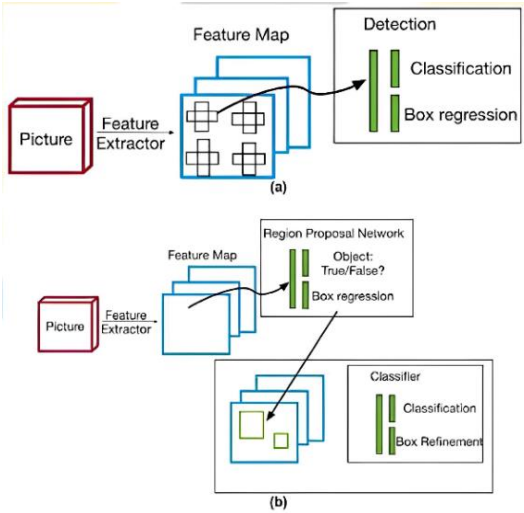
Deteksi objek adalah proses mengidentifikasi atau menemukan contoh dari kelas objek visual tertentu (seperti manusia, hewan, atau kendaraan) dalam sebuah citra digital. Sebagai masalah fundamental dalam bidang visi komputer, deteksi objek bertujuan untuk mengembangkan model komputasi yang memberikan informasi dasar yang diperlukan untuk aplikasi visi komputer, yaitu apa objeknya dan di mana lokasinya berada. Dalam dua dekade terakhir, deteksi objek telah melalui dua era, yaitu deteksi objek pada era tradisional (sebelum tahun 2014) dan deteksi objek berbasis *deep learning* (setelah tahun 2014)(Zou et al., 2023).

Era atau periode deteksi objek tradisional terjadi sebelum ditemukannya representasi citra yang efektif. Pada saat itu, proses deteksi objek masih menggunakan metode tradisional yang memerlukan desain fitur yang kompleks dan berbagai fitur untuk mengoptimalkan komputasi dengan sumber daya yang terbatas. Di sisi lain, model pendeteksian objek berbasis *deep learning* mampu mempelajari representasi fitur yang lebih

kompleks dan canggih. Oleh karena itu, waktu yang tepat untuk menerapkan pendeteksian objek berbasis deep learning adalah ketika proses pendeteksian objek telah terbukti efektif dan efisien. *Deep learning* sendiri merupakan cabang dari machine learning yang memungkinkan komputer mengembangkan perilaku berdasarkan data empiris, seperti data sensor atau basis data, dengan menggunakan beberapa lapisan pemrosesan informasi non-linear untuk melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi(Zhao et al., 2019).

Deteksi objek menggunakan deep learning dapat dibagi menjadi dua kategori, yaitu *One-stage Detect* atau deteksi dalam satu langkah, dan *Two-stage Detect* atau deteksi dalam dua langkah. Dalam deteksi objek satu langkah, proses deteksi objek dapat dilakukan hanya dalam satu langkah. Di sisi lain, dalam deteksi dua langkah, terdapat dua tahap dalam proses deteksi objek. Tahap pertama melibatkan prediksi proposal kandidat objek atau kotak pembatas dari gambar, sementara tahap kedua melibatkan klasifikasi dan regresi pada kandidat kotak pembatas tersebut untuk mengidentifikasi lokasi dan jenis objek(Soviany & Ionescu, 2018).

Meskipun deteksi dua langkah dapat meningkatkan akurasi lokalisasi dan pengenalan objek, hal ini juga memperpanjang waktu yang dibutuhkan untuk melakukan proses deteksi. Sebaliknya, deteksi satu langkah mempercepat proses deteksi, tetapi dengan konsekuensi mengurangi akurasi lokalisasi dan pengenalan objek(Du et al., 2020). Langkah-langkah umum deteksi objek "deteksi satu langkah" dan "deteksi dua langkah" ditunjukkan pada Gambar 2.5.

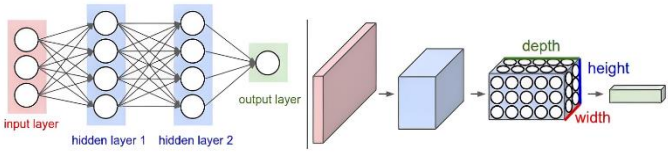


Gambar 2. 5 Plot skema (a) deteksi satu langkah dan (b) deteksi dua langkah
Sumber : (Kemajou et al., 2019)

Pada saat ini, *Convolutional Neural Network* (CNN) telah menjadi standar *de facto* untuk berbagai

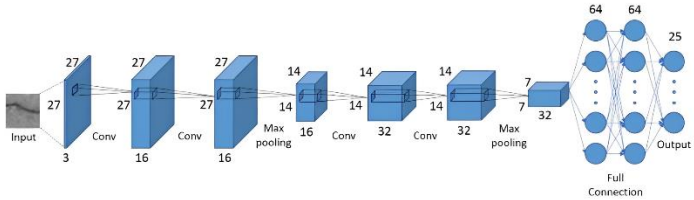
tugas dalam bidang visi komputer. Termasuk di antaranya adalah deteksi objek, klasifikasi gambar, segmentasi, pengenalan wajah, dan teks. Lapisan konvolusi menggabungkan input dan meneruskan hasilnya ke lapisan berikutnya. Konvolusi sendiri merupakan operasi linear, sehingga *Convolutional Neural Network* dapat diartikan sebagai *neural network* yang menggunakan konvolusi setidaknya pada salah satu lapisannya. Setiap bagian kecil (*patch*) pada gambar akan diproses dengan bobot yang sama. Dengan melatih bobot filter konvolusional ini, CNN dapat mempelajari representasi gambar untuk setiap kelas tertentu. Hal ini memungkinkan CNN mengungguli jaringan yang terkoneksi sepenuhnya pada tugas-tugas dalam visi komputer(Danial & Setiawati, 2024).

Metode CNN merupakan pengembangan dari Metode *Multilayer Perceptron* (MLP) yang didesain untuk mengolah data dua dimensi (Pratiwi et al., 2021). Seperti yang terlihat pada Gambar 2.6, cara kerja CNN memiliki kesamaan pada MLP, namun dalam CNN setiap neuron dipresentasikan dalam bentuk tiga dimensi, tidak seperti MLP yang setiap neuron hanya berukuran satu dimensi.



Gambar 2. 6 Dimensi MLP (kiri) dan CNN (kanan)
Sumber : (Pramestya, 2018)

Terdapat beberapa arsitektur CNN yang umum digunakan. Arsitektur tersebut yaitu LeNet, AlexNet, ZF Net, GoogLeNet, VGGNet dan ResNet. Dapat dilihat pada contoh arsitektur CNN pada Gambar , CNN terdiri dari tiga jenis layer, yaitu convolutional layer (Conv), pooling layer (Max pooling) dan fully-connected layer (Full Connection). Tumpukan lapisan tersebut membentuk arsitektur dari CNN



Gambar 2. 7 Contoh Arsitektur CNN
Sumber : (Pramestya, 2018)

Arsitektur dasar *Convolutional Neural Network* (CNN) terdiri dari beberapa jenis lapisan utama. Pertama, terdapat Convolutional Layer, yang menggunakan operasi konvolusi untuk mengekstraksi

fitur-fitur penting dari gambar. Proses ini melibatkan filter dan kernel konvolusi untuk memindai gambar secara bertahap dan menghasilkan representasi yang semakin abstrak. Selanjutnya, hasil dari lapisan konvolusi akan melewati Pooling Layer, yang bertujuan untuk mengurangi dimensi spasial dari representasi yang dihasilkan. Hal ini membantu mengurangi jumlah parameter yang diperlukan dan mempercepat proses komputasi (Eka Putra, 2016).

Setelah itu, hasil dari *Pooling Layer* akan diteruskan ke *Fully Connected Layer*, yang berfungsi sebagai classifier atau regresor. Lapisan ini menghubungkan output dari lapisan konvolusi ke output akhir, dan sering menggunakan fungsi aktivasi seperti ReLU (*Rectified Linear Unit*) atau Leaky ReLU untuk memperkenalkan non-linearitas. *Fully Connected Layer* melakukan klasifikasi atau regresi berdasarkan representasi fitur yang dihasilkan sebelumnya (Suhardin et al., 2021).

Convolutional Neural Network (CNN) telah berhasil diterapkan dalam berbagai aplikasi pengolahan gambar, termasuk klasifikasi objek, deteksi objek, segmentasi, pengenalan wajah, dan banyak lagi. Meskipun arsitektur CNN terus mengalami

perkembangan dan menghadapi tantangan dalam efisiensi, kecepatan, dan akurasi yang lebih baik, secara keseluruhan, CNN telah memberikan kontribusi yang signifikan dalam kemajuan bidang pengenalan pola visual dan pengolahan gambar(Peryanto et al., 2020).

8. YOLO (*You Only Look Once*)

You Only Look Once (YOLO) merupakan salah satu algoritma pendeteksi objek berbasis deep learning yang pertama kali dikembangkan oleh Joseph Redmon pada tahun 2015. YOLO adalah algoritma deteksi objek yang berasal dari pengembangan metode *Convolutional Neural Network* (CNN). Algoritma ini merupakan salah satu algoritma deteksi objek "*One-stage Detector*" pertama yang berbasis *Deep Learning* (Leriansyah & Kurniawardhani, 2020).

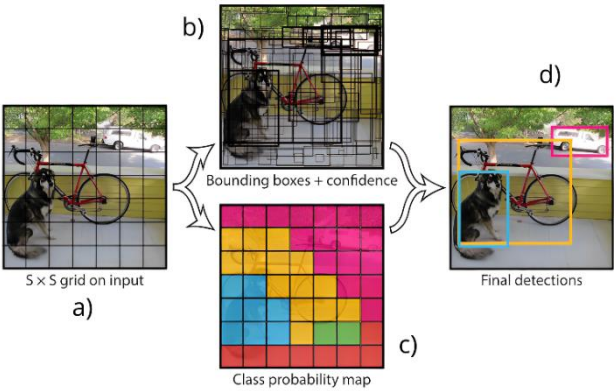
YOLO (*You Only Look Once*) memiliki kemampuan untuk memprediksi objek apa saja yang terdapat dalam sebuah gambar, beserta lokasinya, hanya dengan melihat gambar tersebut satu kali. Cara kerja YOLO didasarkan pada penggunaan jaringan konvolusi yang terus memprediksi bounding box dan probabilitas kelas dari kotak-kotak yang ada pada gambar(Redmon et al., 2016).

YOLO (*You Only Look Once*) memiliki pendekatan yang berbeda dari algoritma pendeteksian objek tradisional. Berbeda dengan paradigma "pendeteksian proposal (kotak pembatas kandidat) dan verifikasi (klasifikasi dan lokalisasi objek)" yang digunakan oleh algoritma tradisional, YOLO hanya menggunakan satu lapisan jaringan syaraf pada gambar. YOLO membagi gambar menjadi beberapa wilayah dan secara bersamaan memprediksi kotak pembatas dan probabilitas pada setiap wilayah (Zou et al., 2023). Meskipun YOLO memiliki kecepatan deteksi yang lebih cepat, namun terkadang menghasilkan lebih banyak kesalahan lokasi objek karena tidak melakukan langkah deteksi proposal terlebih dahulu. Selain itu, YOLO juga menghadapi kesulitan dalam mendeteksi objek yang kecil dan bergerombol.

Algoritma YOLO (You Only Look Once) membagi gambar input menjadi sejumlah sel grid berukuran S x S (kotak kecil)(Ardiansyah et al., 2022). Jika titik pusat objek dalam gambar berada di dalam salah satu sel grid, maka sel grid tersebut akan mendeteksi objek tersebut. Setiap sel grid bertanggung jawab untuk memprediksi jumlah kotak pembatas (*bounding box*) dan nilai kepercayaan (*confidence*) untuk setiap kotak pembatas.

Nilai kepercayaan (*confidence score*) mencakup informasi tentang seberapa yakin model bahwa sebuah objek ada di dalam *bounding box*, serta nilai untuk akurasi estimasi *bounding box* itu sendiri. Gambar 2.8 menunjukan alur kerja algoritma YOLO.

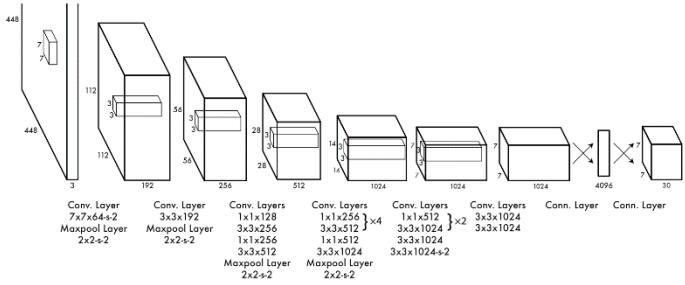
Gambar yolo (a) adalah proses Yolo membagi gambar masukan menjadi kotak 7 x 7. Bagian (b) adalah hasil prediksi *bounding box* hasil pengolahan. Bagian (c) menunjukkan prediksi kelas objek, dan (d) adalah hasil prediksi objek beserta nilai *confidence* serta kelas objek.



Gambar 2. 8 Alur kerja algoritma YOLO dalam pendeteksian objek pada citra
Sumber : (Redmon et al., 2016)

YOLO (*You Only Look Once*) bekerja berdasarkan prinsip "*single shot*", yang berarti bahwa jaringan arsitektur diatur sedemikian rupa sehingga dalam satu

lintasan bingkai (*frame*) dapat mendeteksi beberapa objek secara serentak. Sistem deteksi YOLO menggunakan input berupa gambar atau video dan menggabungkan komponen-komponen terpisah ke dalam satu jaringan saraf untuk memprediksi bounding box. Algoritma YOLO dapat melakukan prediksi objek berupa bounding box B dan skor kepercayaan C dengan membagi input gambar atau video menjadi grid berukuran SxS. Setiap sel grid kecil bertanggung jawab untuk memprediksi jika titik tengah suatu objek jatuh pada grid tersebut. Bounding box tersebut mengandung lima elemen prediksi, yaitu x, y, w, h, dan skor kepercayaan C. Koordinat x dan y mewakili titik tengah dari kotak pembatas (bounding box) relatif terhadap batas grid. Elemen w (lebar) dan h (tinggi) dianggap relatif terhadap keseluruhan gambar(Redmon et al., 2016).



Gambar 2. 9 Arsitektur YOLO
Sumber : (Redmon et al., 2016)

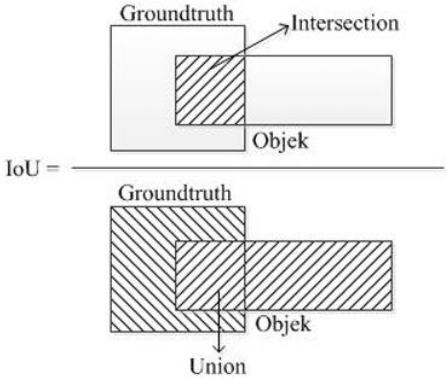
Setiap sel grid akan memprediksi kotak pembatas (Bounding) B dan kepercayaan C. Skor kepercayaan C ini mencerminkan apakah objek tersebut berada pada kotak tersebut dan juga seberapa akurat kotak itu dalam memperkirakan. Prediksi dari nilai kepercayaan C dinyatakan dengan IoU dengan rumus

$$confident = P_r(Object) \times IOU \frac{truth}{pred} \quad (2.1)$$

IOU merupakan singkatan dari *Intersection over Union*, merupakan sebuah matrik evaluasi untuk mengukur tingkat keakuratan dalam mendeteksi sebuah objek pada dataset. Metrik ini bekerja dengan cara menghitung luas area pertemuan antara kotak prediksi objek dan kotak kebenaran dasar (ground truth), kemudian membaginya dengan luas area gabungan dari kedua kotak tersebut. IOU memberikan gambaran seberapa baik model dapat mendeteksi objek secara akurat, dengan nilai IOU yang lebih tinggi menunjukkan tingkat keakuratan yang lebih baik. IOU dirumuskan pada persamaan

$$IOU = \frac{A \cap B}{A \cup B} \quad (2.2)$$

Dimana $A \cap B$ adalah luas area pertemuan dua kotak, dan $A \cup B$ adalah luas area gabungan dari kedua kotak.



Gambar 2. 10 Ilustrasi perhitungan IOU
Sumber : (Pramestya, 2018)

Sejak Algoritma YOLO pertama kali di kenalkan pada tahun 2015 algoritma ini mengalami beberapa perkembangan diantaranya adalah(Jiang et al., 2022):

a. YOLOv2

   Dikembangkan pada tahun 2016 yang menambahkan *anchor* dengan *K-means*, *Two-stage Training*, serta *Full Convolutional Network.*

b. YOLOv3

   Dikembangkan pada tahun 2018 yang menambahkan deteksi *Multi-scale* dengan menggunakan FPN.

c. YOLOv4.

   Dikembangkan pada tahun 2019 yang menambahkan SPP, fungsi aktivasi MISH, *Data*

Enhancement Mosaic/Mixup, GIOU (*Generalized Intersection over Union*) *Loss Function.*

d. YOLOv5

Dikembangkan pada tahun 2020 yang menambahkan kontrol yang lebih fleksibel terhadap ukuran model, pengaplikasian fungsi aktivasi *Hardswish*, dan *Data Enhancement.*

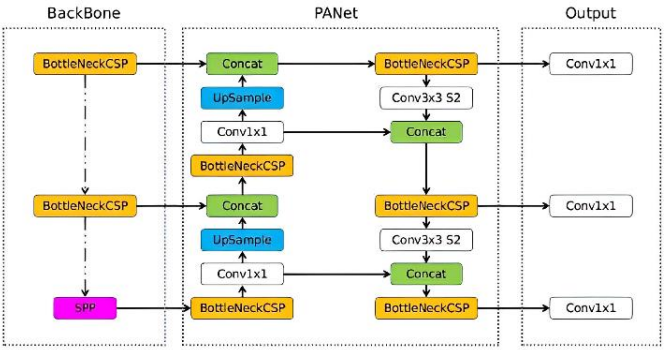
Selain beberapa versi YOLO yang telah disebutkan di atas, ada juga beberapa versi YOLO lainnya yang memiliki kelebihan dan kekurangan masing-masing. Beberapa di antaranya adalah YOLO 9000, Tiny YOLO, Fast YOLO, dan YOLACT (*You Only Look at Coefficients*).

9. YOLOv5

YOLOv5 adalah kerangka kerja deteksi objek dengan algoritma yang dikembangkan pada tahun 2020 oleh Glenn Jocher, seorang peneliti dan CEO Ultralystics LLC. YOLOv5 menggunakan *framework* PyTorch dan ditulis dalam bahasa pemrograman Python(Fang et al., 2021). Menurut situs web Roboflow, YOLOv5 merupakan hasil pengembangan implementasi YOLOv3 dari framework PyTorch yang juga dikembangkan oleh Glenn Jocher. YOLOv5 memiliki lima model yang telah dilatih sebelumnya

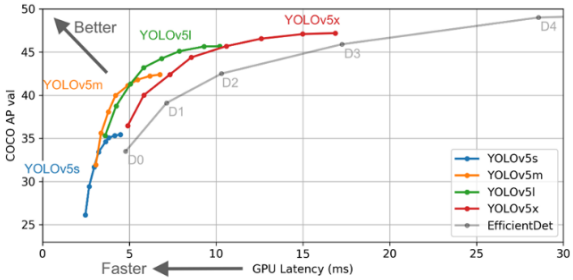
dengan ukuran berbeda, yaitu YOLOv5s (ukuran terkecil), YOLOv5m, YOLOv5l, dan YOLOv5x (ukuran terbesar).

Secara keseluruhan, arsitektur YOLOv5 terdiri dari 3 bagian utama yaitu *backbone, neck*, dan *head*. Pada YOLOv5 dilakukan berbagai teknik augmentasi data untuk meningkatkan kemampuan model dalam melakukan generalisasi dan mengurangi *overfitting*. Teknik-teknik augmentasi tersebut terdiri dari *mosaic augmentation, copy-paste augmentation, random affine transformations, mix-up augmentation, albumentations, HVS augmentation, dan random horizontal flip*. Selain itu, diterapkan juga beberapa strategi training yang dapat meningkatkan performa model dalam melakukan deteksi seperti, *multiscale training, autoanchor, warmup and cosine LR scheduler, exponential moving average, mixed precision training, dan hyperparameter evolution* (Zhou et al., 2021). Pada training menggunakan YOLOv5, nilai *loss* atau kerugian dikomputasikan menggunakan tiga kombinasi komponen yang terdiri dari *classes loss, objectness loss, dan location loss.*



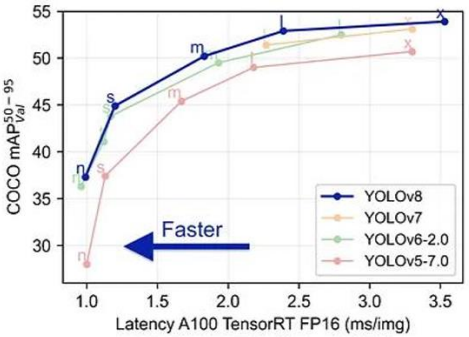
Gambar 2. 11 Network Arsitektur Yolov5
Sumber : (Hidayat, 2023)

Gambar 2.11 emberikan ilustrasi tentang bagaimana arsitektur YOLOv5 memproses data dengan lebih efisien, terutama dalam hal konsumsi dataset, yang jauh lebih kecil dibandingkan dengan penggunaan jaringan saraf konvolusional (Convolutional Neural Network atau CNN) yang tidak mengimplementasikan YOLO. Efisiensi ini menjadikan YOLOv5 sebagai salah satu versi paling canggih dari keluarga YOLO, yang dirancang untuk mempercepat proses deteksi objek secara signifikan tanpa mengorbankan akurasi deteksi. (Aini et al., 2021). Beberapa informasi yang terkandung dalam gambar meliputi *cross-stage partial networks* (CSP), *partial pyramid pooling* (SPP), *convolutional layers* (Conv), dan *concatenate functions* (Concat).



Gambar 2. 12 Perbandingan performa kerja pada tiap model YOLOv5
Sumber : (Cahyani, 2023)

Dengan menggabungkan berbagai fitur baru, peningkatan, dan strategi pelatihan, YOLOv5 melampaui versi sebelumnya dari keluarga YOLO dalam hal performa dan efisiensi.Berdasarkan Gambar 2.12, waktu inferensi yang dibutuhkan oleh YOLOv5 dalam melakukan deteksi berkisar pada rentang waktu 5 ms hingga 20 ms.



Gambar 2. 13 Performa YOLOv5 dengan YOLO versi terbaru
Sumber : (Cahyani, 2023)

Gambar 2.13 menunjukkan grafik perbandingan waktu inferensi yang dibutuhkan dan akurasi mAP_0.5:95 dalam melakukan deteksi menggunakan algoritma YOLOv5, YOLOv6, YOLOv7, dan YOLOv8. Berdasarkan grafik, YOLOv8 unggul dalam nilai akurasi dibandingkan YOLO versi lainnya. Akan tetapi, YOLOv5 lebih cepat dalam kecepatan deteksi, karena membutuhkan waktu inferensi yang lebih rendah dibandingkan YOLO versi lainnya dengan rentang waktu 1 ms hingga 3.25 ms(Cahyani, 2023).

10. Metrik Evaluasi Multiclass

Ada beberapa metode pengujian kinerja yang dapat mengukur keberhasilan suatu jaringan, salah satunya adalah *multiclass confusion matrix*. Matriks evaluasi merupakan suatu organisasi prediksi dari masalah klasifikasi(Amwin, 2021). *Confusion matrix*, juga dikenal sebagai matriks kerancuan atau matriks kesalahan, memberikan informasi tentang sejauh mana hasil klasifikasi yang dihasilkan oleh sistem (model) sesuai dengan hasil klasifikasi yang sebenarnya. Matriks ini menggambarkan kinerja model klasifikasi pada set data uji, di mana nilai aktualnya telah diketahui. *Multiclass confusion matrix* membandingkan Jumlah prediksi yang benar untuk kelas tertentu (*true*

positive) dengan Jumlah prediksi salah di mana model memprediksi kelas tersebut, padahal kelas sebenarnya berbeda (*false positive*), dan jumlah prediksi salah di mana model tidak memprediksi kelas yang benar. (*false negative*) dibandingkan dengan kejadian yang tidak terjadi (*false positive*). Tabel 2.1 menunjukan contoh multiclass confusion matrix yang digunakan untuk mengukur kelas A, B, dan C.

Tabel 2. 1 Multiclass Confusion Matrix

Confusion matrix		Actual Class		
		A	B	C
Predicted Class	A	TP_A	FP_B_A	FP_C_A
	B	FP_A_B	TP_B	FP_C_B
	C	FP_A_C	FP_B_C	TP_C

Dimana TP_A adalah jumlah data yang benar-benar kelas A dan diprediksi A, FP_B_A adalah jumlah data yang sebenarnya kelas A, tetapi diprediksi kelas B, FP_C_A adalah jumlah data yang sebenarnya kelas A, tetapi diprediksi kelas C dan seterusnya.

Matriks confusion memiliki empat istilah yang perlu dipahami: *True positive* (TP): Model memprediksi dengan benar dan hasilnya memang benar. *True negative* (TN): Model memprediksi dengan salah dan

hasilnya memang salah. *False positive* (FP): Model memprediksi dengan benar, padahal hasilnya sebenarnya salah. *False negative* (FN): Model memprediksi dengan salah, padahal hasilnya sebenarnya benar.

Tidak ada nilai true negatif yang dapat ditemukan karena nilai-nilai ini berasal dari klasifikasi yang benar secara keseluruhan untuk kejadian yang sebenarnya tidak terjadi. Namun, banyak kejadian yang seharusnya tidak terdeteksi pada gambar malah diklasifikasikan sebagai negatif(Hidayat, 2023). Pada tabel 2.2 adalah rumus-rumus dalam *confusion matrix* yang dapat digunakan untuk menghitung nilai akurasi, presisi, recall, dan F1 score.

Tabel 2. 2 Perhitungan Metrik

No	Metrik	Persamaan
1	*Accuracy*	$\frac{TP + TN}{TP + FP + TN + FN}$
2	*Precision*	$\frac{TP}{TP + FP}$
3	*Recall*	$\frac{TP}{TP + FN}$
4	*F1-Score*	$2 \times \frac{Precision \times recall}{Precision + recall}$

Akurasi (*Accuracy*) adalah perbandingan antara jumlah prediksi yang benar (positif dan negatif) dengan total data. *Recall* mengukur sejauh mana model dapat mengidentifikasi data positif dari total data positif yang sebenarnya. Presisi (*Precision*) mengukur sejauh mana prediksi positif yang benar dari total prediksi positif. *F1-Score* adalah rata-rata harmonis antara presisi dan recall. Nilai *F1-Score* berkisar antara 0 hingga 1, dengan nilai tertinggi menunjukkan hasil yang baik. *F1-Score yang* baik menandakan bahwa model klasifikasi memiliki keseimbangan yang baik antara presisi dan *recall.*

Dalam penelitian selain menggunakan *confusion matrix* peneliti juga menggunakan mAP (*mean average precision*) untuk melakukan evaluasi objek deteksi yang dilakukan oleh sistem. Metrik evaluasi mAP (*mean average precision*) umumnya digunakan untuk mengevaluasi model *machine learning* dalam tugas deteksi objek, seperti Fast R-CNN, YOLO, dan Mask R-CNN. Nilai mAP dihitung berdasarkan nilai presisi dan recall, dan berada dalam rentang 0 hingga 1. Formula untuk menghitung mAP melibatkan beberapa *sub-metrik*, termasuk *confusion matrix, intersection over union* (IoU), *recall*, dan *precision*.

Beberapa tahapan dalam menghitung average precision (AP) adalah sebagai berikut:

1) mengetahui nilai prediksi dari model.
2) konversi nilai prediksi ke label kelas klasifikasi.
3) menghitung nilai confusion matrix – TP, TN, FP dan FN.
4) menghitung metrik precision dan recall.
5) menghitung nilai bawah area pada kurva precision – recall.
6) mengukur average precision.

Melalui tahapan tersebut dapat dihitung nilai mAP dengan mencari nilai average precision (AP) masing – masing kelas dan kemudian rata – rata dengan jumlah kelas yang ada (Harani et al., 2019).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} APi \qquad (2.3)$$

11. Python

Python adalah bahasa pemrograman yang sering digunakan dalam pengembangan situs web, perangkat lunak, ilmu data, dan machine learning. Keunggulan Python terletak pada efisiensinya, kemudahan dalam pembelajaran, serta kemampuannya untuk berjalan di berbagai platform. Karena sifatnya yang serbaguna,

Python menjadi salah satu bahasa pemrograman yang populer di kalangan pengembang (Cahyani, 2023).

Python adalah bahasa pemrograman yang menggunakan pendekatan berorientasi objek untuk mengeksekusi sejumlah instruksi secara langsung (dalam mode interpretasi). Python memiliki semantik maju yang memastikan tingkat keterbacaan sintaksis yang tinggi. Bahasa ini menggabungkan sintaks kode yang sangat mudah dibaca dengan kekuatan pustaka yang luas dan komprehensif. Meskipun Python merupakan bahasa pemrograman tingkat lanjut, namun dirancang agar mudah dipelajari dan dipahami (Gunawan & Santoso, 2021).

12. Google Colab

Google Colab atau Google Colaboratory adalah dokumen yang dapat dieksekusi yang memungkinkan pengguna untuk menyimpan, membuat, dan berbagi program melalui Google Drive. Layanan ini disediakan secara gratis dan juga mendukung unit pemrosesan grafis (GPU) gratis. Google Colab dapat digunakan untuk meningkatkan kemampuan dalam bahasa pemrograman Python serta mengembangkan aplikasi deep learning dengan menggunakan pustaka-pustaka

populer seperti Keras, TensorFlow, PyTorch, dan OpenCV (Hidayatulloh, 2021).

Colab Notebooks memudahkan pengguna untuk menggabungkan kode yang dapat dieksekusi (bahasa mesin) dengan teks kaya dalam satu dokumen. Pengguna dapat menggunakan gambar, HTML, LaTex, dan banyak lagi(Kuroki, 2021). Hasil dari Colab Notebooks akan otomatis disimpan di akun Google Drive. Selain itu, pengguna dapat berkolaborasi dengan orang lain dengan membagikan Colab Notebooks.

13. Roboflow

Roboflow adalah platform yang menyediakan layanan untuk membangun aplikasi *computer vision* kepada pengembang yang diluncurkan pada tahun 2020. Roboflow menyederhanakan proses dalam pemberian label (*labeling*) pada data hingga pelatihan model (*training*)(Alexandrova et al., 2015). Pengembang dapat menggunakan Roboflow untuk melakukan beberapa hal sebagai berikut:

a. Anotasi citra atau mengunggah anotasi yang telah ada.
b. Melakukan konversi file anotasi VOC XML ke COCO JSON.
c. Melakukan pengecekan pelabelan.

d. Melakukan pre-processing citra.
e. Meningkatkan kualitas citra sebagai data untuk dilatih: flip, rotate, brighten/darken, chop, shear, blur, dan add random noise.
f. Menghasilkan format anotasi seperti: TFRecords, Create ML dan Turi Create, dan custom YOLOv5 implementations.
g. Memudahkan dalam mengakses kualitas datasets.
h. Mendapat dan membagikan dataset umum.

14. Flask

Flask adalah sebuah web framework ringan yang ditulis dalam bahasa pemrograman Python. Framework ini digunakan untuk membangun aplikasi web dengan cara yang cepat dan efisien. Flask memungkinkan pengembang untuk dengan mudah membuat aplikasi web yang skalabel, dengan dukungan untuk berbagai fitur seperti *routing* URL, mengelola permintaan dan tanggapan HTTP, serta integrasi dengan berbagai ekstensi untuk memperluas fungsionalitasnya. Kelebihan Flask antara lain kesederhanaan, fleksibilitas, dan dokumentasi yang baik, membuatnya menjadi pilihan populer bagi pengembang web Python (Mufid et al., 2019).

B. KAJIAN PENELITIAN YANG RELEVAN

Penelitian ini tentunya membutuhkan rujukan dari penelitian lain sebagi bahan informasi dan acuan tambahan guna mendukung penelitian ini terselesaikan. Beberapa penelitian yang sudah dilakukan yang relevan dengan penelitian ini adalah sebagai berikut :

Tabel 2. 3 Kajian Penelitian relevan

No	Nama Penulis	Judul Penelitian	Hasil Kajian
1	Bandi Sasmito, dkk (2023)	Deteksi Kerusakan Jalan Menggunakan Pengolahan Citra Deep Learning di Kota Semarang	Penelitian ini menggunakan prinsip penginderaan jauh dengan teknologi Jaringan Syaraf Tiruan Deep Learning. YOLO (You Only Look Once) versi 4 digunakan untuk deteksi kerusakan jalan. Hasil pendeteksian ditambahkan posisi atau lokasi dengan menggunakan Global

			Navigation Satellite System (GNSS), sehingga nantinya hasil deteksi dapat memberikan posisi atau lokasi yang akurat. Penelitian ini menghasilkan model identifikasi kerusakan jalan dengan nilai overall accuracy sebesar 88% dan nilai kappa accuracy sebesar 86% dan lokasi sebaran kerusakan yang memiliki koordinat posisi dengan akurasi RMSE sebesar ± 5,6 meter
2	Dwi Arief Adityah (2021)	Deteksi dan Klasifikasi Keretakan Jalan	Pada penelitian ini memanfaatkan pengolahan citra digital menggunakan

		Menggunakan Metode You Only Look Once	metode You Only Look Once versi-3 sebagai langkah awal bagi pengurus jalan untuk membantu menentukan jenis kerusakan jalan retak. Metode YOLO-V3 adalah metode yang memperkenalkan Darknet-53, yang merupakan fitur ekstraktor lebih detail dari versi sebelumnya sehingga dapat mendeteksi objek dengan detail dan cepat. Penelitian ini memperoleh hasil pengujian dengan metode yang digunakan dapat mengklasifikasi

			lubang dengan tingkat akurasi sebesar 96,4 % dari total 1107 data.
3	Eko Bagus Yanuar (2023)	Sistem Deteksi Lubang Dan Keretakan Jalan Pada Mobil Otonom Menggunakan Algoritma Convolutional Neural Network	Pada penelitian ini, metode pengenalan objek menggunakan algoritma CNN (Convolutional Neural Network) pada YoloV5 dan YoloV3. Penulis menggunakan algoritma tersebut untuk membedakan jalan berlubang dan tidak berlubang dengan akurasi, kecepatan deteksi objek, dan kecepatan pemrosesan gambar yang tinggi. Model Yolov5 mengungguli model Yolov3 dengan

			perbedaan akurasi untuk data validasi 0,1354 dan data uji 0,44. Model yolov5m dengan jumlah layer 291 menghasilkan akurasi data validasi 0.87.
4	Bima Putra Gusti Pamungkas, dkk (2021)	Deteksi Dan Menghitung Manusia Menggunakan YOLO-CNN	penelitian ini menggunakan YOLOv3 dan YOLOv2 yang merupakan salah satu algoritma dari Deep Learning network sebagai metode pembelajaran mendalam untuk mendeteksi dan menghitung manusia. Hasil yang diperoleh dengan menggunakan model YOLOv3 mendapatkan nilai

			rata-rata confidence 0,90 dibandingkan dengan YOLOv2 yang memiliki rata-rata nilai confidence 0,61. Hal ini dikarenakan YOLOv3 memiliki jumlah layer lebih banyak dibandingkan dengan YOLOv2. Namun, waktu yang diperlukan dalam menjalankan program dengan YOLOv3 lebih lama dibandingkan dengan YOLOv2.
5	Aldhityatika Amwin (2021)	Deteksi Dan Klasifikasi Kendaraan Berbasis Algoritma You Only Look Once (Yolo)	Dalam penelitian ini penulis menggunakan algoritma YOLO untuk melakukan pendeteksian dan klasifikasi kendaraan.

			Penelitian ini menggunakan dataset sebanyak 531 gambar dengan lima kelas yaitu mobil, sepeda motor, truk, bus, dan becak. Hasil penelitian menunjukan algoritma You Only Look Once (YOLO) dapat mengenali objek pada video CCTV yang dipasang di Simpang Air Mancur-Imanuel Kota Medan dengan menggunakan pre-trained weights yang telah dilatih sendiri dengan nilai mean Average Precision (mAP) sebesar 99,35%.

Dari kajian seperti ditunjukkan pada tabel 2.1 diatas, maka penelitian terkait Implementasi Model *Deep Learning* dalam Deteksi dan Klasifikasi Jenis Kerusakan Jalan Aspal dengan Algoritma *You Only Look Once* (YOLO) V5, memiliki topik yang sangat menarik untuk dibahas dikarenakan belum terlalu banyak penelitian yang membahas tentang algoritma YOLOv5 dalam mendeteksi kerusakan jenis jalan. Beberapa penelitian tersebut dapat dijadikan sumber dan referensi bagi penulis dalam melaksanakan penelitian ini.

Adapun perbedaan penelitian ini dengan penelitian sebelumnya yaitu pada penelitian ini model deteksi yang nantinya dilatih akan diimplementasikan pada website yang dibangun menggunakan framework flask. Sehingga hasil deteksi dapat diakses secara langsung melalui antarmuka web, yang mana akan memudahkan pengguna dalam mengakses fitur deteksi tanpa memerlukan instalasi perangkat lunak tambahan.

BAB III

METODE PENELITIAN

A. Metode Pengumpulan Data

Pada penelitian ini data yang akan digunakan adalah Data Primer dan Data Sekunder. Data primer adalah data yang dikumpulkan dan diperoleh dengan cara langsung dengan terjun ke lapangan seperti survei, wawancara, kuisoner dan sebagainya. Sedangkan Data Sekunder adalah data yang diperoleh atau dikumpulkan dari sumber-sumber yang telah ada.

Data Primer yang digunakan pada penelitian ini diperoleh dari pengambilan data berupa gambar kerusakan jalan secara langsung oleh peneliti di kota semarang, Sedangkan untuk Data Sekunder diperoleh dari beberapa sumber yang telah ada seperti dari jurnal, ataupun dari internet.

B. Kebutuhan Perangkat Penelitian

Pada pelaksanaan penelitian ini tentunya dibutuhkan perangkat yang dapat menunjang keperluan penelitian. Perangkat yang dibutuhkan dibagi menjadi dua yaitu perangkat keras dan perangkat lunak. Pada tabel 3.1 dideskripsikan perangkat apa saja yang akan digunakan pada penelitian ini.

Tabel 3. 1 Alat dan bahan penilitian

Perangkat Keras			
No	Nama	Spesifikasi	Keterangan
1	Laptop	Intel® Core™ i3-1215U, RAM 8.00 GB	Perangkat utama yang akan digunakan untuk pembuatan dan pengujian project.
2	Handphone (kamera)	Camera 64MP	Perangkat yang akan digunakan untuk pengambilan data objek berupa gambar secara langsung di lapangan.
Perangkat Lunak			
No	Nama	spesifikasi	Keterangan
1	Python	Versi 3.12.0	Bahasa pemrograman yang digunakan dalam pembuatan sistem deteksi.
2	YOLO	YOLOv5	Algoritma deteksi objek yang

			digunakan pada project.
3	Flask	Versi 3.0.2	*Framework* web yang digunakan untuk pembuatan GUI.
4	Google Colaboration		Code editor yang digunakan untuk melatih data dan mendapatkan model terlatih.
5	Roboflow		Web yang digunakan untuk melakukan preprocessing dan image labeling pada dataset.
6	Visual Studio Code	Versi 1.89.1	Code editor yang digunakan dalam pembangunan sistem project.

C. Metode Penelitian

Terdapat beberapa tahapan yang terdapat dalam penelitian ini, di antaranya adalah pengambilan data dan pengumpulan dataset, praroses dataset, labelling dataset, training model, testing model, pembuatan sistem deteksi dan implementasi model.

1. Pengumpulan Dataset

Pada tahap pengambilan dan pengumpulan dataset, peneliti mengumpulkan gambar atau citra yang diperlukan untuk membangun atau melatih jaringan. *Dataset* diperoleh dari pengambilan data secara langsung oleh peneliti di lapangan serta berbagai sumber dari internet. Untuk Jumlah dataset yang akan dikumpulkan yaitu sebanyak 580 dengan kerusakan jalan berupa lubang (pothole), retak halus (hair cracking), dan retak kulit buaya (alligator crack).

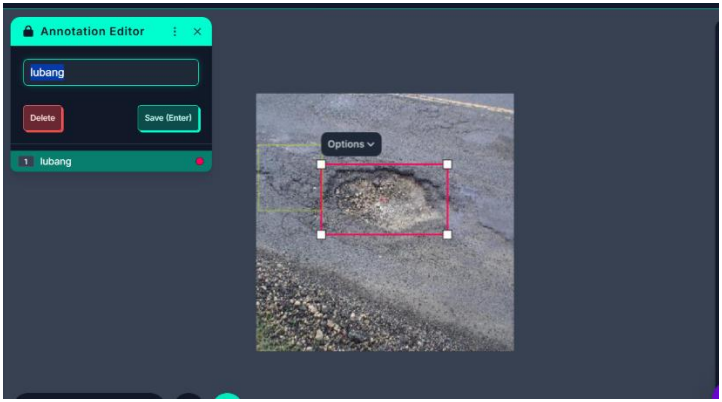
2. Praproses Dataset

Pada tahap ini data yang diperoleh akan diolah terlebih dahulu, yaitu dengan merubah ukuran tiap gambar agar sama satu sama lain dengan resolusi 640x640, kemudian data gambar akan diaugmentasi. Augmentasi sendiri merupakan proses transformasi citra secara acak, dalam penelitian ini transformasi citra yang digunakan adalah rotasi (perputaran) dan

refleksi (pencerminan). Augmentasi digunakan untuk meningkatkan jumlah data *training* dan mengurangi *overfitting.*

3. Labeling Dataset

Setelah dilakukan pemrosesan data gambar, langkah selanjutnya adalah melakukan pelabelan. Pelabelan bertujuan untuk memberikan tanda pengenal atau identitas pada setiap kelas objek dalam gambar yang telah dilabeli. Proses pelabelan melibatkan pembuatan bounding box dan penulisan nama kelas pada setiap objek. Pelabelan dataset mencakup informasi tentang kelas objek, koordinat x dan y, serta ukuran panjang dan lebar objek. Pelabelan dataset dilakukan secara manual dengan menggunakan platform *Roboflow*.



Gambar 3. 1 Contoh Pelabelan Gambar pada Roboflow

4. **Training Model**

Pada tahap ini dataset yang telah dilabeli akan dilatih menjadi model deteksi dengan melibatkan beberapa parameter tertentu, agar model dapat mengenali objek dengan akurasi tinggi. Pada tahap ini model akan dilatih dengan menggunakan Google Colaboration.

5. **Testing Model**

Pada tahap ini akan dilakukan pengujian dengan untuk mengukur tingkat akurasi dari sistem deteksi dan klasifikasi jenis kerusakan jalan. Pengujian dilakukan dengan menggunakan metode perhitungan akurasi, diantaranya adalah dengan menggunakan nilai mAP (*Mean Average Presicion*) dan *Confusion Matrix.*

6. **Pembuatan Sistem Deteksi**

Sistem deteksi yang dibuat pada penelitian ini akan dibuat menggunakan *framework* Flask, Sehingga hasilnya akan ditampilkan dalam bentuk webapp. Berikut adalah tahapan pada pembuatan sistem deteksi ini :

a. Pembuatan GUI

GUI atau *Grapichal User Interface* adalah sebuah tampilan antarmuka grafis pada komputer yang memungkinkan pengguna berinteraksi

dengan perangkat lunak melalui elemen visual. Pada pembuatan sistem deteksi ini GUI dibuat sesederhana mungkin dengan beberapa fungsi utama yaitu pilih gambar dari device dan upload gambar untuk dideteksi dan kemudian gambar hasil deteksi akan ditampilkan dan disimpan dalam folder lokal.

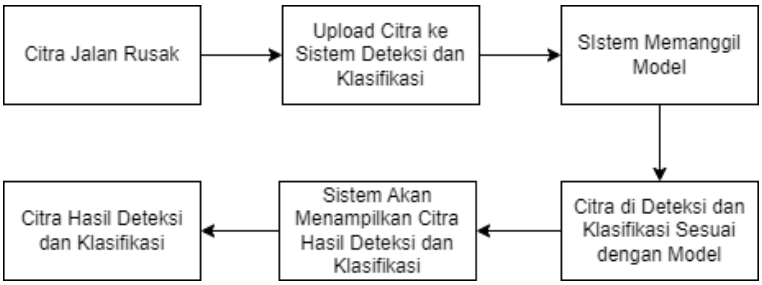
Gambar 3. 2 Desain GUI

b. Penulisan Kode Program

Pada tahap ini sistem akan dibuat sesuai dengan desain yang telah dibuat. Penulisan kode program menggunakan *Text Editor Visual Studio Code.* Bahasa pemrograman yang digunakan adalah Python dengan tambahan HTML untuk membuat tampilan web.

c. Implementasi Model

Pada tahap ini model yang telah dilatih sebelumnya akan diimplementasikan pada sistem deteksi, sehingga sistem dapat melakukan deteksi dan klasifikasi pada kerusakan jenis jalan sesuai dengan model yang telah dilatih.



Gambar 3. 3 Blok Diagram Sistem

d. Pengujian Sistem deteksi

Setelah Sistem deteksi berhasil dibangun tahap selanjutnya yaitu dengan melakukan pengujian pada sistem deteksi untuk mengetahui apakah sistem dapat melakukan deteksi sesuai dengan yang diharapkan serta untuk mengetahui berapa total waktu yang diperlukan oleh sistem untuk melakukan deteksi pada kerusakan jalan aspal.

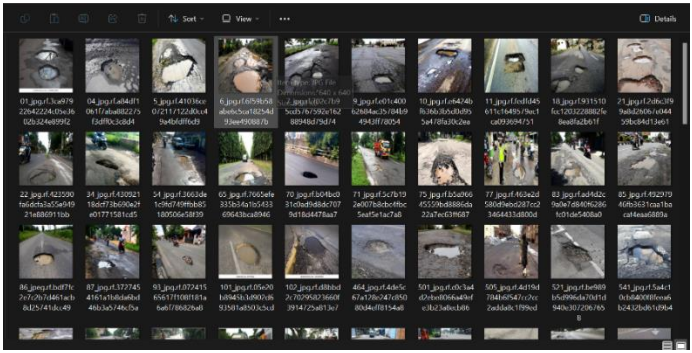
BAB IV

HASIL DAN PEMBAHASAN

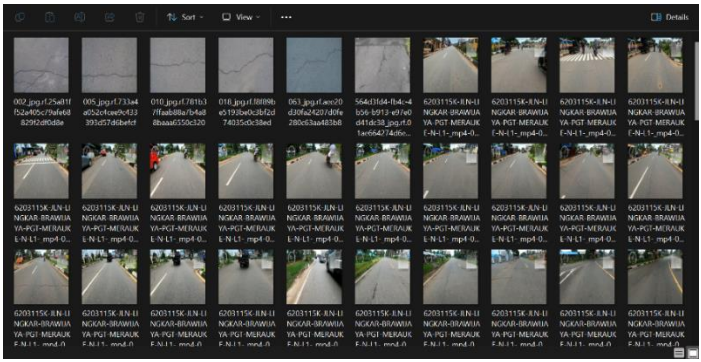
Pada Bab IV ini mendeskripsikan tentang langkah-langkah pembuatan model deteksi kerusakan jalan aspal dengan algoritma Yolov5, pembuatan sistem deteksi, serta implementasi model kedalam sistem deteksi kerusakan jalan aspal.

A. Pengumpulan Dataset

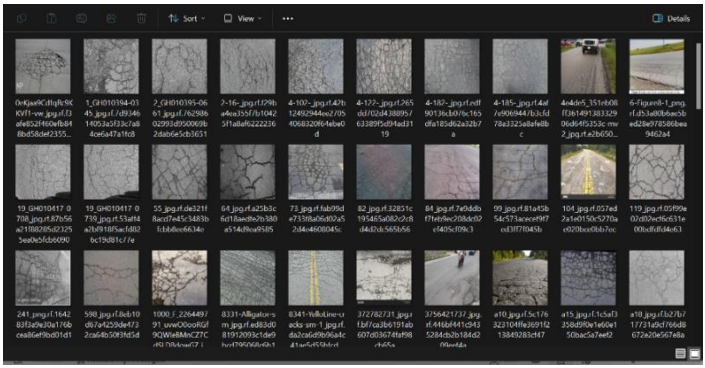
Dataset yang digunakan pada penelitian ini diambil dari beberapa sumber diantaranya seperti dari google maps dan web roboflow universe yang mana berisikan dataset publik. Kemudian dataset berupa gambar tersebut penulis kumpulkan sesuai dengan jenis kerusakan jalan yang akan diteliti. Total dataset yang diperoleh adalah 580 gambar untuk 3 kelas kerusakan jalan.



Gambar 4. 1 Data kerusakan jalan lubang



Gambar 4. 2 Data kerusakan jalan retak halus



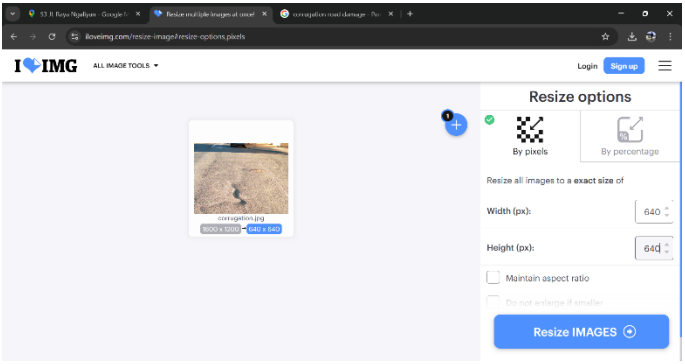
Gambar 4. 3 Data kerusakan jalan retak kulit buaya

B. Praproses Data

Pada tahapan sebelumnya dataset yang dikumpulkan masih berupa data mentah yang dikumpulkan dari beberapa sumber yang berbeda. Pada tahap ini dataset berupa gambar tersebut akan diolah agar sesuai dengan kebutuhan penelitian ini. Pengolahan dataset pada penelitian ini adalah sebagai berikut

1. Menyamakan Ukuran Data Gambar

Pada tahap ini ukuran data gambar yang digunakan akan disamakan menjadi ukuran 640x640 dengan tujuan untuk memastikan bahwa ukuran gambar yang akan masuk ke dalam model akan seragam, sehingga akan memudahkan pemrosesan. Selain itu dengan menyamakan ukuran data gambar akan meningkatkan efisiensi dalam pelatihan serta meningkatkan kompabilitas dengan arsitektur YOLOv5, dikarenakan dengan ukuran gambar yang tidak serupa akan membuat arsitektur model perlu disesuaikan secara dinamis sehingga akan meningkatkan komplesitas pembuatan model.



Gambar 4. 4 Merubah ukuran gambar menjadi 640x640

2. Augmentasi Data

Augmentasi Data merupakan proses yang sangat penting dalam pembuatan model YOLOv5. Dikarenakan pada proses ini akan meningkatkan

keragaman data secara dinamis pada dataset sehingga akan meningkatkan variasi dan kualitas pada data training. Selain itu Augmentasi data juga bertujuan untuk meningkatkan kemampuan dan performa model dalam mendeteksi objek dalam berbagai kondisi. Dari dataset yang telah dikumpulkan akan diaugmentasi dengan cara rotasi dan pencerminan secara vertikal dan horizontal (*vertikal flip* dan *horizontal flip*). Berikut ini adalah contoh data sebelum dan setelah di augmentasi.

Gambar 4. 5 Lubang sebelum di augmentasi

Gambar 4. 6 Horizontal Flip

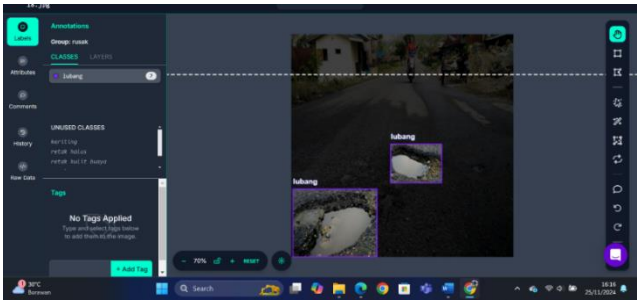
Gambar 4. 7 Vertikal Flip

Gambar 4. 8 Rotasi 180°

C. Labeling Dataset

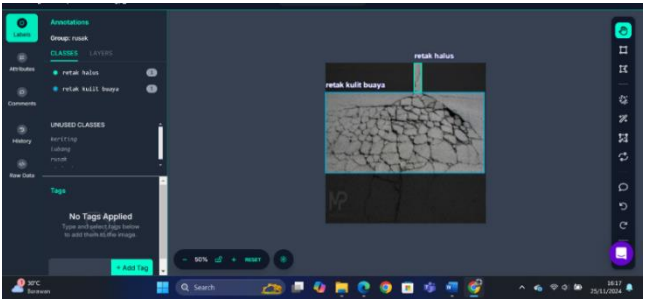
Tahap ini merupakan proses pemberian label pada data gambar dengan cara memberikan *bounding box* (kotak batas) beserta dengan nama kelas pada masing-masing objek disetiap data gambar. Proses labeling dataset pada pembuatan model deteksi jenis kerusakan jalan aspal ini dengan memanfaatkan *tool* pada situs *roboflow.com*. Pelabelan terdiri dari 3 nama kelas yaitu lubang, retak halus dan retak kulit buaya. Pelabelan dengan

menggunakan *tool* dari situs *roboflow.com* dapat dilihat pada gambar berikut.



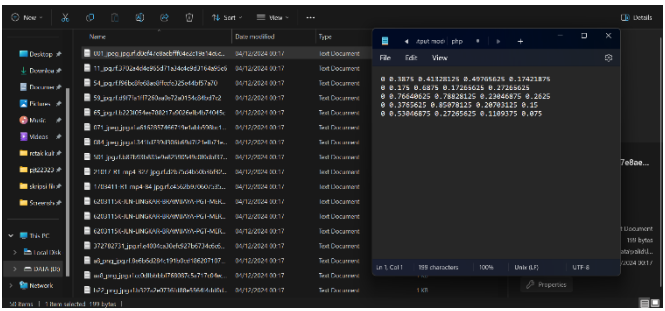
Gambar 4. 9 Pelabelan data kelas Lubang

Gambar 4. 10 Pelabelan data kelas Keriting



Gambar 4. 11 Pelabelan data kelas Retak Kulit Buaya

Gambar yang telah diberi label akan menghasilkan file berekstensi .TXT yang akan dibaca sebagai anotasi atau label YOLO, sehingga sistem pada saat melakukan training model deep learning akan mudah untuk membaca dan memahaminya.



Gambar 4. 12 format dan isi file label

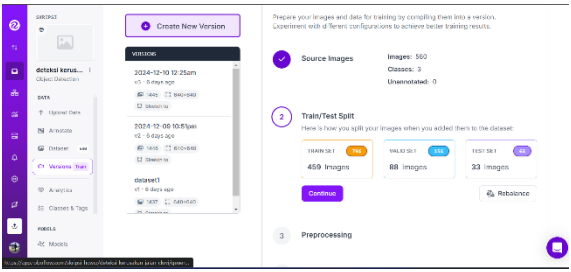
D. Training Model

Pada tahap ini data gambar yang telah diberi label akan dilatih menjadi model dengan proses deep learning. Proses training data ini dilakukan dengan menggunakan google collab. Tahap training bertujuan untuk menciptakan model yang dapat mengenali dan mempelajari suatu pola dalam pengenalan deteksi dan klasifikasi objek sesuai dengan class yang diinginkan serta dengan tingkat akurasi yang tinggi.

1. Persiapan Training

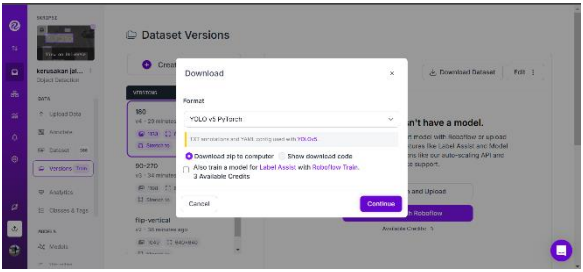
Hal yang perlu dilakukan sebelum melakukan training model adalah menyiapkan tools dan data yang

dibutuhkan saat proses training model. Yang harus dilakukan pertama kali adalah Membuat versi baru dataset serta membagi jumlah data menjadi data training, data valid, dan data testing. Untuk pembagiaannya data train sebanyak 459, data valid sebanyak 88 dan data test sebanyak 33. Setelah dibagi dataset dapat dibuat dengan menekan button create pada bagian bawah.

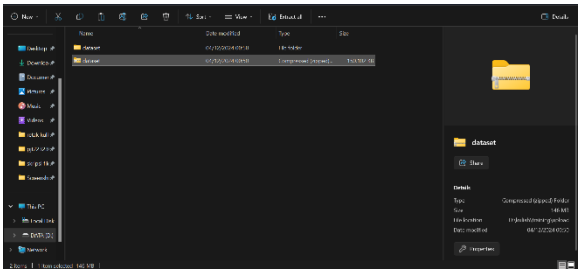


Gambar 4. 13 Create version dataset

Di roboflow yang akan digunakan mengunduh dataset yang telah diberi label di platform roboflow dengan format YOLO v5 PyTorch.

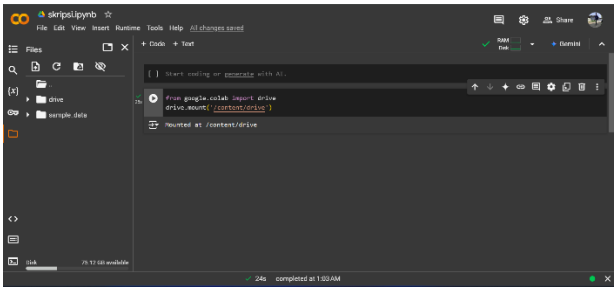


Gambar 4. 14 Donwload dataset yang telah dilabeli



Gambar 4. 15 Hasil download dataset

Langkah selanjutnya upload folder dataset dalam format zip ke dalam google drive. Kemudian buka platform Google Collab yang nantinya akan digunakan untuk proses training lalu hubungkan Google Drive yang berisi dataset.

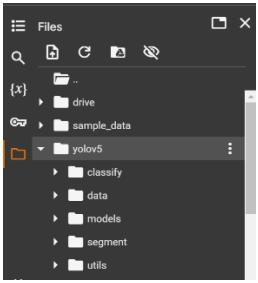


Gambar 4. 16 Menghubungkan Colab dengan GDrive

Langkah berikutnya buat perintah clone repository ultralytics YOLOv5 dan menginstal dependencies-nya. Jika proses cloning berhasil maka akan muncul folder yolov5 pada navbar yang ada pada sebelah kiri Googgle Colab.

```
!git clone https://github.com/ultralytics/yolov5  # clone
!cd yolov5
!pip install -r requirements.txt  # install
```

Gambar 4. 17 Clone Repository dan dependency YOLOv5

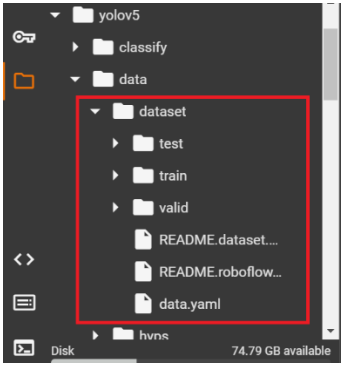


Gambar 4. 18 Hasil Clone repository YOLOv5

Langkah selanjutnya ekstrak file dataset dari Google Drive dan simpan folder hasil ekstrak ke dalam directory yolov5/data.

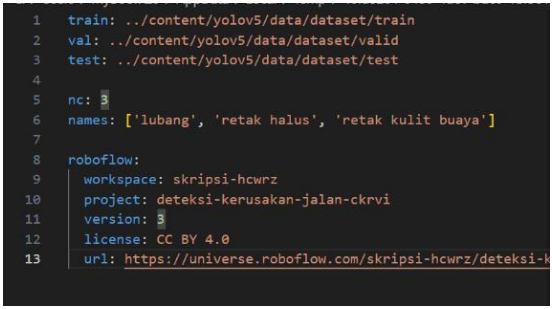
```
!unzip /content/drive/MyDrive/download/dataset.zip -d /content/yolov5/data
```

Gambar 4. 19 Perintah unzip file dataset



Gambar 4. 20 Folder dataset

Setelah file dataset berhasil di ekstrak maka langkah berikutnya adalah mengubah path folder valid train dan test yang ada pada file data.yaml pada directory yolov5/data/dataset. Hal ini bertujuan untuk menunjukan dimana letak folder train, valid, dan test.



Gambar 4. 21 File data.yaml

2. Proses Training

Setelah langkah-langkah persiapan training dilalui, langkah selanjutnya adalah proses training dataset menjadi model deteksi dengan menggunakan proses deep learning. Tulis perintah training model pada Google Collab.

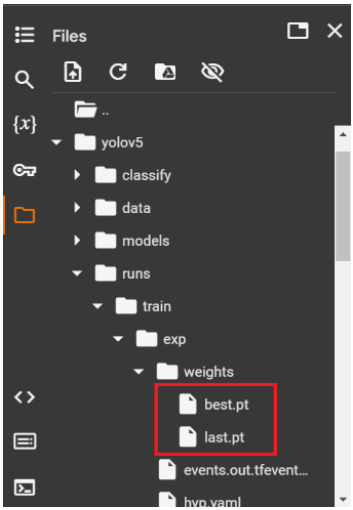
Gambar 4. 22 Perintah untuk training dataset

Pada perintah training diatas terdapat beberapa parameter yang berpengaruh pada proses training yaitu --img, --batch, --epoch, --data, --weights. Perintah !python train.py bertujuan untuk memanggil

dan mengeksekusi file train.py pada folder yolov5. "--img" bertujuan untuk mengatur ukuran image yang digunakan dalam proses training, pada penelitian ini digunakan ukuran 640. Parameter "--batch" berfungsi untuk membagi keseluruhan data menjadi beberapa bagian sesuai dengan batch yanb diinput, dalam penelitian ini batch yang digunakan sebanyak 16 batch. Pembagian ini bertujuan untuk meringankan beban kinerja GPU saat melakukan proses training. Kemudian parameter "--epoch" berfungsi untuk mengatur seberapa banyak data akan dilatih. Semakin besar nilai epoch maka kualitas model hasil training akan semakin baik. Dalam penelitian ini epoch yang digunakan bernilai 100. "--data" bertujuan untuk memanggil file data.yaml yang telah diubah sebelumnya. Sedangkan parameter "--weight" berfungsi untuk menentukan versi yolov5 apa yamg akan digunakan dalam melatih model. Semakin tinggi tingkatan weight maka hasil deteksi akan semakin baik namun kemampuan GPU untuk melatih dan mendeteksi objek akan semakin berat. Dalam penelitian ini versi yolov5 yang digunakan adalah yolov5s.pt.

Setelah dataset berhasil dilatih maka folder akan disimpan dalam direktori "/yolov5/run/train" dalam

folder train akan ada folder exp yang mana folder ini akan berisikan model yang telah berhasil dilatih, model dalam folder ini akan ada 2 dengan nama best.pt dan last.pt yang mana best.pt adalah model terbaik dari keseluruhan epoch saat proses training model dan last.pt adalah model pada epoch terakhir saat proses training.



Gambar 4. 23 Hasil training model

E. Testing Model

Tahap ini bertujuan untuk mengevaluasi kinerja model YOLOv5 dalam mendeteksi objek secara akurat serta memastikan integrasi dengan framework Flask berjalan optimal. Pengujian dilakukan melalui serangkaian

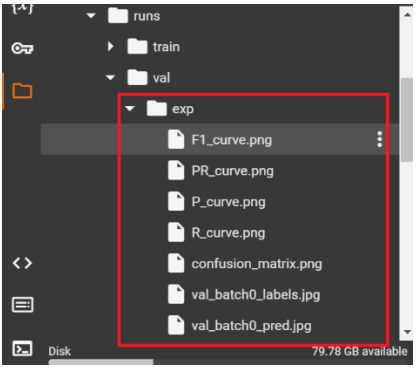
eksperimen yang melibatkan dataset uji dan simulasi penerapan pada aplikasi berbasis web.

1. Testing Berbasis Skrip

Untuk mengetahui seberapa baik model yang telah dilatih maka perlu dilakukan testing pada model. Untuk melakukan testing model dapat menggunakan perintah berikut pada Google colab.

Gambar 4. 24 Perintah untuk melakukan testing model

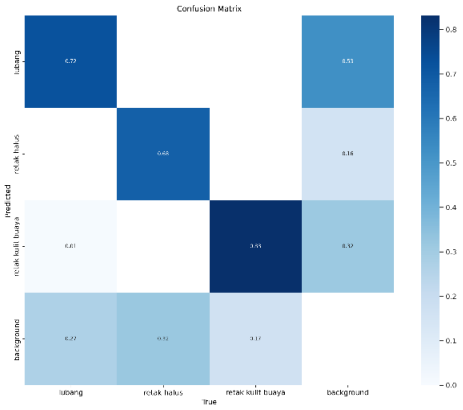
Perintah tersebut akan mengeksekusi program val.py yang mana merupakan program yang dapat digunakan untuk melakukan testing dan evaluasi pada model. Untuk parameter "--weigths" digunakan untuk mengidentifikasi model mana yang akan diuji dengan best.pt adalah model yang akan diuji. Untuk parameter "--data" digunakan untuk memanggil dataset yang mana data.yaml berisikan path folder dataset. Setelah perintah tersebut dijalankan maka hasil test akan disimpan pada ./yolov5/runs/val/exp dalam folder tersebut akan berisi beberapa file yang dapat digunakan untuk melihat hasil testing model.



Gambar 4. 25 Hasil evaluasi model

a. Confusion matrix

Hal pertama yang harus dilihat saat ingin mengetahui hasil testing model adalah dengan melihat confusion matrix.



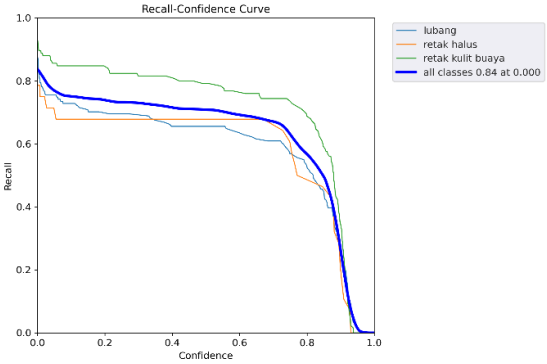
Gambar 4. 26 Gambar Confusion Matrix hasil testing

Berdasarkan gambar di atas untuk model best.pt memiliki confusion matrix yang cukup baik

dengan class lubang memiliki nilai true positif yang cukup bagus yaitu 0.72, untuk class retak halus 0.68 dan class retak kulit buaya 0.83.

b. Recall

Recall merupakan salah satu parameter yang digunakan untuk mengetahui apakah model yang telah dilatuh sudah baik atau tidak. Pada model best.pt rata-rata dapat mendeteksi semua class dengan nilai recall 0.84 pada confidece threshold 0.000.

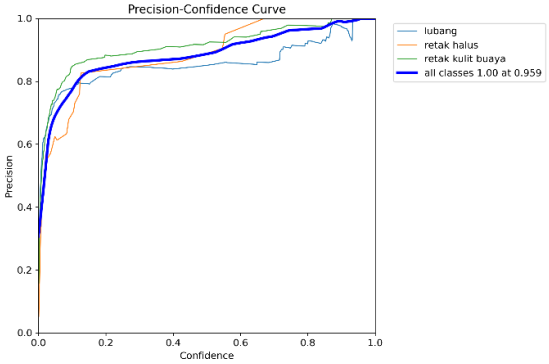


Gambar 4. 27 Gambar Curva Recall hasil evaluasi

c. Presicion Skor

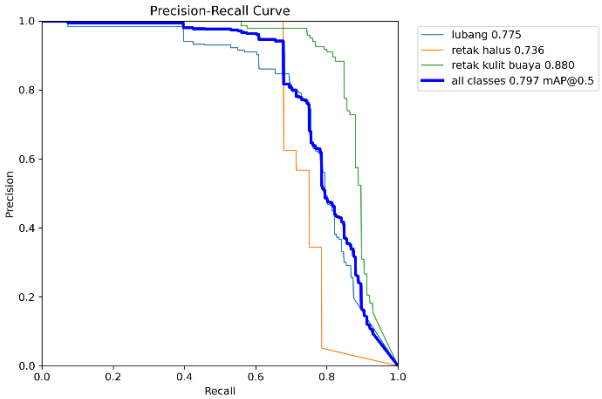
Untuk kurva presisi berkebalikan dengan kurva recall yang mana kurva presisi yang semakin naik menunjukan kualitas model yang semakin baik. Pada model best.pt kurva mengarah keatas

dengan rata-rata semua class mencapai 1 pada nilai confidence 0.959. Yang berarti model akan membuat prediksi dengan sangat yakin jika nilai confidence berada diatas 0.959 (95.9%) yang mana semuanya akan akurat.



Gambar 4. 28 Gambar curva Precision-Confidence

d. Precision-Recall Skor

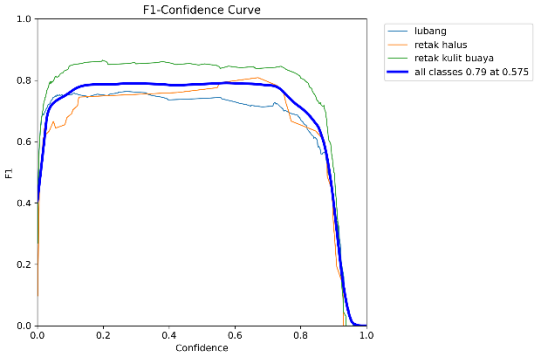


Gambar 4. 29 Gambar Curva Precision-Recall

Setelah melihat Recall dan Precision maka selanjutnya kedua parameter tersebut perlu dibandingkan untuk melihat nilai skor Presicion-recall. Pada model best.pt didapatkan skor untuk tiap classnya yaitu 0.075 untuk class lubang, 0.736 untuk class retak halus dan 0.880 untuk class retak kulit buaya.

e. F1-Skor

F1-skor merupakan parameter penentu dalam evaluasi kinerja model klasifikasi dengan menggabungkan metrik Precision dan Recall. F1-skor sangat berguna untuk menyeimbangkan kedua metrik Presicion dan Recall.



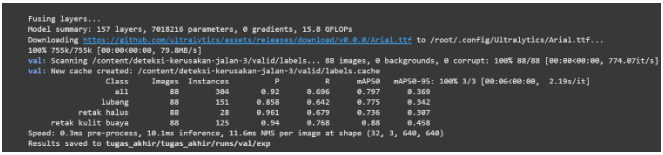
Gambar 4. 30 Gambar Curva F1-Confidence

Pada model best.pt memiliki F1-skor yang cukup baik yang mana secara keseluruhan “all classes 0.79 at 0.575” berarti model akan

memberikan F1-Skor sebesar 0.79 untuk seluruh class dalam dataset ketika ambang batas probabilitas ditetapkan pada 0.575.

f. mAP (mean Average Precision)

Untuk mengetahui nilai mAP pada model bisa dilihat pada bagian output di Google Collab setelah menjalankan perintah untuk melakukan evaluasi model sebelumnya.



Gambar 4. 31 Output testing pada Google Colab

Pada gambar tersebut dapat dilihat hasil evaluasi model best.pt. Berdasarkan hasil tersebut dapat dilihat nilai AP (*Average Precision*) untuk *class* lubang sebesar 0.775, *class* retak halus sebesar 0.736 dan *class* retak kulit buaya sebesar 0.88. Maka untuk nilai mAP (mean Average Precision) untuk semua class yang ada pada model deteksi adalah 0.797.

2. Testing Berdasarkan Input Gambar

Langkah berikutnya yaitu menguji model secara manual dengan menggunakan data input berupa

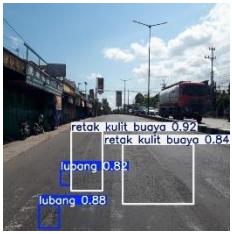
gambar. Untuk melakukan uji model dapat menggunakan perintah berikut pada google colab.

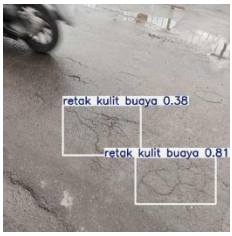
```
[5]  !python detect.py --weights /content/best.pt --img 640 --conf 0.25 --source /content/testing/testing
     # display.Image(filename='runs/detect/exp/zidane.jpg', width=600)
```

Gambar 4. 32 Perintah untuk mendeteksi gambar

Perintah detect.py merupakan salah satu program dalam YOLOv5 yang berfungsi untuk mendeteksi objek sesuai dengan model yang digunakan. Parameter "--weights" digunakan untuk model yang akan dideteksi dan "--source" menunjukan sumber input gambar yang akan dideteksi.

Tabel 4. 1 Tabel hasil uji model pada Google Collab

No	Gambar	Hasil
1		Nilai confidence tertinggi ada pada kelas retak kulit buaya sebesar 0.92 dan terendah ada pada kelas lubang 0.82
2		Nilai confidence sebesar 0.77 pada kelas retak halus

3		Nilai confidence kelas retak kulit buaya sebesar 0.72 dan 0.68 pada kelas lubang
4		Nilai confidence sebesar 0.85 pada kelas lubang
5		Terdapat 3 objek dengan 2 kelas retak halus yang memiliki nilai confidence masing-masing 0.84 dan 0.62
6		Terdeteksi 2 objek kelas retak kulit buaya dengan nilai confidence masing-masing sebesar 0.81 dan 0.38

7		Nilai confidence sebesar 0.94 pada kelas lubang
8		Nilai confidence kelas retak kulit buaya sebesar 0.87 dan 0.63 pada kelas lubang
9		Nilai confidence kelas retak kulit buaya sebesar 0.92 dan 0.88 pada kelas lubang
10		Nilai confidence kelas retak kulit buaya sebesar 0.85 dan 0.90 pada kelas lubang

Berdasarkan hasil uji coba menggunakan input gambar pada tabel diatas menunjukan bahwa model

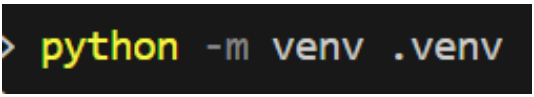
dapat mendeteksi jenis kerusakan jalan lubang, retak halus dan retak kulit buaya dengan baik.

F. Pembuatan Sistem Deteksi

Proses berikutnya setelah model berhasil dilatih adalah membuat sistem deteksi guna menampilkan dan mungaplikasikan model dalam mendeteksi kerusakan jalan secara langsung. Sistem deteksi yang dibuat berbasis web dan memiliki tampilan yang sederhana dengan fitur upload gambar dan secara langsung akan menampilkan hasil deteksi kerusakan jalan sesuai dengan model yang telah dilatih pada tahap sebelumnya. Pembuatan sistem ini akan memanfaatkan framework flask dalam pengembangannya

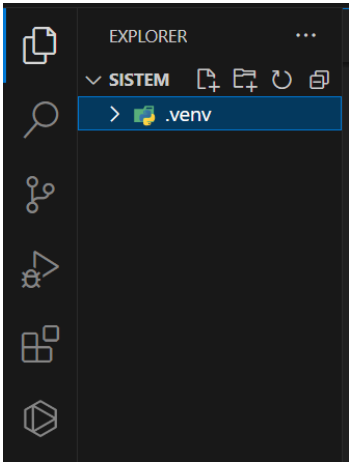
1. Persiapan environment flask

Langkah pertama yaitu menyiapkan virtual environment untuk pengembangan sistem. Yaitu dengan cara masuk ke folder projek melalui vs code, setelah itu buka terminal dari vs code, kemudian tulis perintah berikut python -m venv .venv pada terminal.



Gambar 4. 33 Membuat Virtual Environment pada Python

Perintah tersebut akan membuat virtual environment dengan nama .venv. Jika berhasil maka akan muncul folder seperti gambar berikut.



Gambar 4. 34 Tampilan setelah Virtual Environment berhasil dibuat

Langkah berikutnya setelah venv berhasil dibuat yaitu mengaktifkan venv dengan cara menjalankan perintah berikut pada terminal.

```
PS D:\kuliah\sistem> .venv\Scripts\activate
(.venv) PS D:\kuliah\sistem>
```

Gambar 4. 35 Mengaktifkan Virtual Environment

Kemudian install flask pada virtual environment dengan menuliskan perintah berikut.

```
PS D:\kuliah\sistem> .venv\Scripts\activate
(.venv) PS D:\kuliah\sistem> pip install flask
```

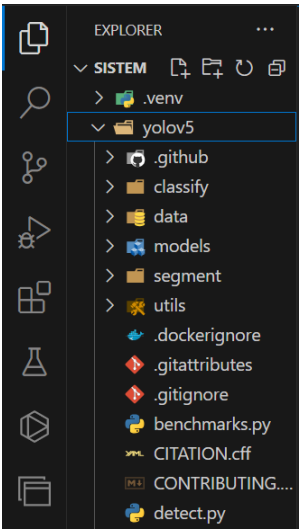
Gambar 4. 36 Menginstall Flask pada Python

Setelah venv berhasil diaktifkan dan flask berhasil diinstall. Langkah berikutnya *clone repository* YOLOv5 dan *install dependencies* yang diperlukan dengan menuliskan perintah berikut pada terminal.

```
git clone https://github.com/ultralytics/yolov5  # clone
cd yolov5
pip install -r requirements.txt  # install
```

Gambar 4. 37 Clone Repository dan install dependency untuk YOLOv5

Jika perintah tersebut berhasil maka akan muncul folder yolov5 pada folder projek yang dibuat serta secara otomatis akan mendownload dependencies yang diperlukan untuk menjalankan YOLOv5.

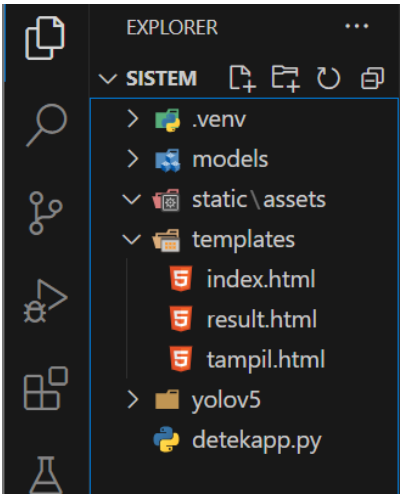


Gambar 4. 38 Tampilan setelah Repository YOLOv5 berhasil di clone

Setelah langkah-langkah diatas dilalui maka persiapan environment untuk membuat website menggunakan flask sudah berhasil.

2. Penulisan Kode Program

Pada tahap ini hal pertama yang perlu dilakukan adalah membuat struktur file yang akan digunakan pada sistem deteksi.

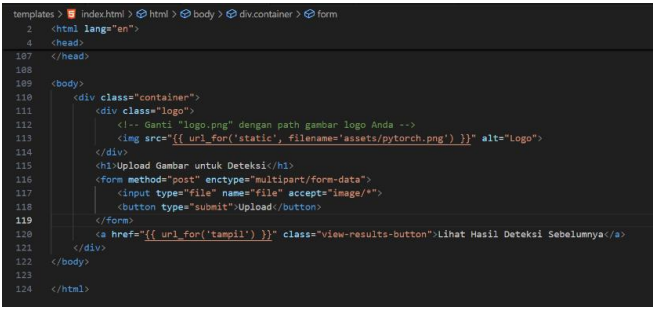


Gambar 4. 39 Struktur folder dan file pada pembuatan sistem deteksi

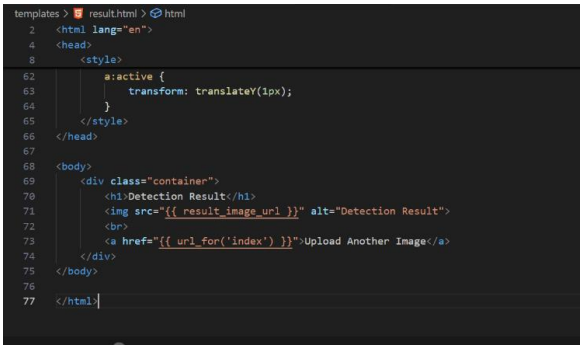
Pada folder models akan berisi file model deteksi yang telah dilatih dengan metode *deep learning* file model yang digunakan akan berekstensi .pt atau file pytorch. Folder static akan berisi folder assets selain itu juga berguna untuk menyimpan gambar hasil deteksi. Folder templates akan berisikan file html yang akan

menjadi tampilan website dengan, index.html sebagai halaman utama, result.html untuk menampilkan gambar setelah dideteksi dan tampil.html untuk menampilkan gambar hasil deteksi sebelumnya. Sedangkan file detekapp.py adalah file utama yang berfungsi untuk mengatur semua proses pada halaman website yang dibuat.

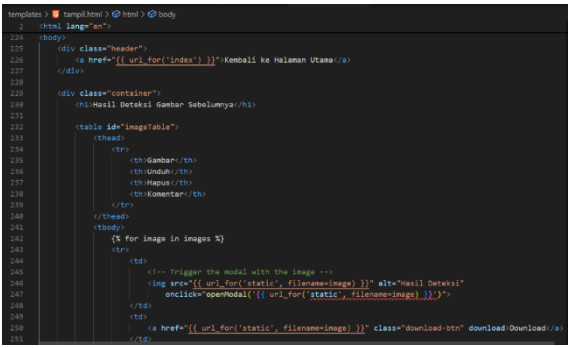
Langkah berikutnya yaitu menulis kode program pada file-file yang telah dibuat dimulai dari file html.



Gambar 4. 40 Coding untuk halaman index.html

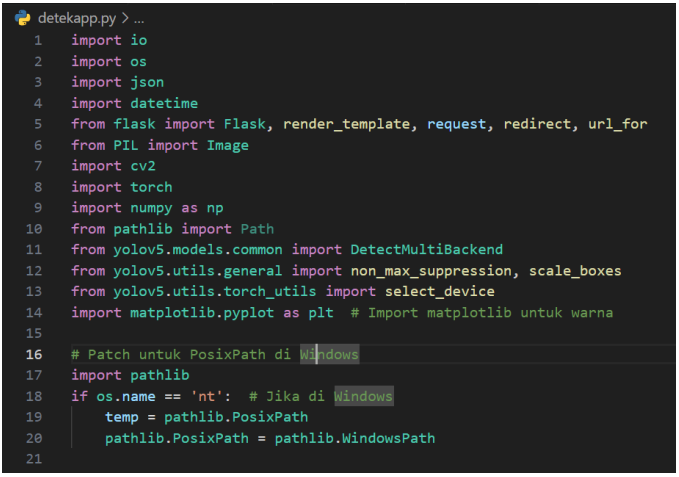


Gambar 4. 41 Coding untuk halaman result.html



Gambar 4. 42 Coding untuk halaman tampil.html

Setelah file html dibuat langkah berikutnya yaitu menuliskan kode program pada file detekapp.py file ini akan berisikan seluruh alur kerja website serta fungsi-fungsi yang diperlukan dalam melakukan deteksi kerusakan jalan. Pada bagian awal tulis program untuk memanggil library yang dibutuhkan.



Gambar 4. 43 Import library yang dibutuhkan

Kemudian tuliskan program untuk inisialisasi flask serta buat kode untuk memuat model.

```python
# Inisialisasi Flask
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = 'static'
DATETIME_FORMAT = "%Y-%m-%d_%H-%M-%S-%f"

# Load model YOLOv5
MODEL_PATH = "models/"
device = select_device('CPU')  # Pilih perangkat (GPU jika tersedia, atau CPU)
model = DetectMultiBackend(MODEL_PATH, device=device)
names = model.names
img_size = 640  # Resolusi input gambar
```

Gambar 4. 44 Inisialisasi projek Flask

Setelah itu buat fungsi untuk menggambar bounding box serta menempelkan label pada gambar hasil deteksi.

```python
def draw_boxes(image, boxes, names, colors):
    """
    Menggambar bounding box dengan label yang sesuai panjang teks.
    """
    for *box, conf, cls in boxes:
        cls = int(cls)  # Pastikan kelas adalah integer
        x1, y1, x2, y2 = map(int, box)
        label = f"{names[cls]} {conf:.2f}"
        color = colors[cls]  # Warna unik untuk setiap kelas

        # Gambar bounding box dengan ketebalan 3
        cv2.rectangle(image, (x1, y1), (x2, y2), color, 3)

        # Ukuran teks label
        font_scale = 0.6
        font_thickness = 2
        (label_width, label_height), baseline = cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX, font_scale, font_thickness)

        # Padding untuk latar belakang label
        padding = 5
        label_x2 = x1 + label_width + 2 * padding
```

Gambar 4. 45 Fungsi untuk menggambar bounding box

```python
    # Padding untuk latar belakang label
    padding = 5
    label_x2 = x1 + label_width + 2 * padding
    label_y1 = max(0, y1 - label_height - baseline - 2 * padding)  # Pastikan tidak keluar dari gambar
    label_y2 = y1

    # Gambar latar belakang label
    cv2.rectangle(image, (x1, label_y1), (label_x2, label_y2), color, -1)

    # Tambahkan label teks di atas latar belakang
    text_x = x1 + padding
    text_y = y1 - padding - baseline
    cv2.putText(image, label, (text_x, text_y), cv2.FONT_HERSHEY_SIMPLEX, font_scale, (255, 255, 255), font_thickness)

return image
```

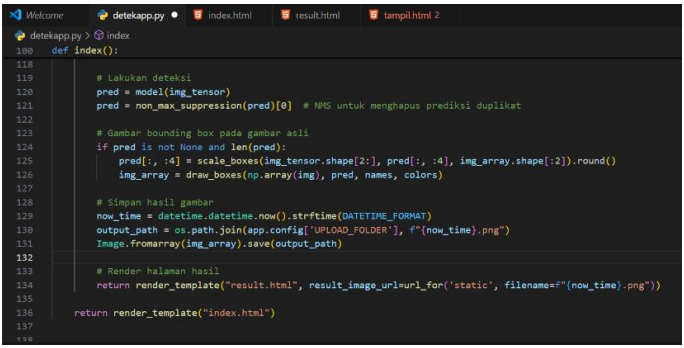
Gambar 4. 46 Fungsi untuk menggambar bounding box

Langkah selanjutnya tulis beberapa fungsi berikut pada detekapp.py.

Gambar 4. 47 Fungsi untuk mendapatkan nama gambar, komentar dan menyimpan komentar

Kemudian buat route untuk halaman index.html yang mana memiliki beberapa fitur yaitu pilih dan upload gambar untuk langsung dideteksi oleh sistem dan nantinya akan ditampilkan dalam halaman result.html.

Gambar 4. 48 Route untuk halaman index.html

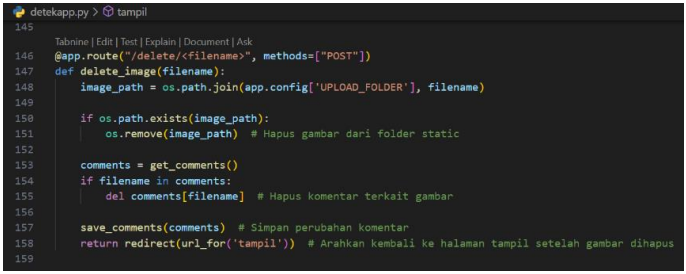


Gambar 4. 49 Route untuk halaman index.html

Langkah selanjutnya buat route untuk menampilkan gambar yang telah dideteksi sebelumnya serta menampilkan komentar terkait gambar.

Gambar 4. 50 Route untuk halaman tampil.html

Kemudian buat route untuk menghapus gambar serta komentar pada gambar yang akan dihapus.



Gambar 4. 51 Route untuk menghapus gambar & komentar

Langkah selanjutnya buat route untuk menambahkan atau menghapus komentar pada gambar hasil deteksi.

```
160
    Tabnine | Edit | Test | Explain | Document | Ask
161 @app.route("/add_comment/<filename>", methods=["POST"])
162 def add_comment(filename):
163     comment = request.form.get('comment')
164     if comment:
165         comments = get_comments()
166         if filename not in comments:
167             comments[filename] = []
168         comments[filename].append(comment)
169         save_comments(comments)  # Simpan komentar ke dalam file JSON
170
171     return redirect(url_for('tampil'))  # Arahkan kembali ke halaman tampil setelah komentar ditambahkan
172
```

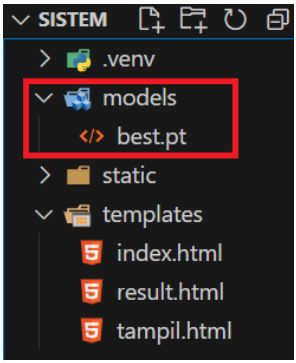
Gambar 4. 52 Route untuk menghapus komentar saja

3. Implementasi Model

Pada tahap ini model yang telah diperoleh dari hasil training akan diimplementasikan pada website sistem deteksi. Pertama-tama unduh model terlebih dahulu di Google Collab pada folder /yolov5/train/exp/weights. Setelah diunduh pindahkan file model tersebut kedalam folder models pada projek sistem deteksi. Kemudian tambahkan juga nama model pada bagian load model di file detekapp.py guna mendapatkan file model di folder models.

```
27  # Load model YOLOv5
28  MODEL_PATH = "models/best.pt"
29  device = select_device('CPU')  # Pilih perangkat (GPU jika tersedia, atau CPU)
30  model = DetectMultiBackend(MODEL_PATH, device=device)
31  names = model.names
32  img_size = 640  # Resolusi input gambar
33
```

Gambar 4. 53 menambahkan path untuk model best.pt pada detekapp.py



Gambar 4. 54 Menaruh file best.pt pada folder models

Setelah model ditambahkan kedalam sistem deteksi maka projek flask dapat langsung di run dengan menuliskan perintah berikut pada terminal.

```
(.venv) PS D:\kuliah\sistem> python detekapp.py
```

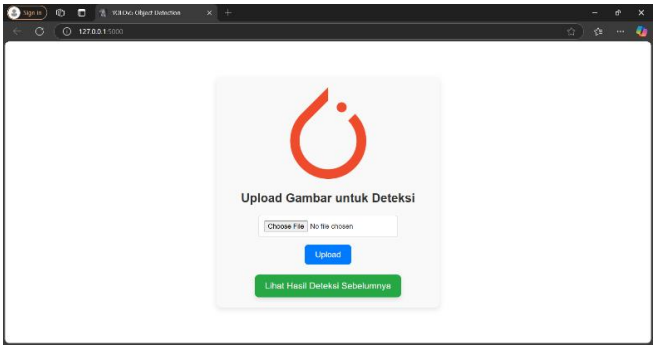
Gambar 4. 55 Perintah untuk menjalankan projek

Kemudian tekan tekan ctrl-klik pada alamat IP dari hasil menjalankan detekapp.py.

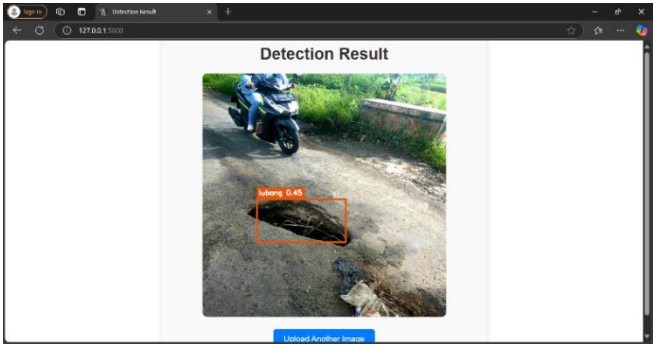
```
 * Debug mode:
WARNING: This Follow link (ctrl + click) ver. Do not use it in a production
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
YOLOv5  2024-12-5 Python-3.12.0 torch-2.3.1+cpu CPU
```

Gambar 4. 56 Alamat IP website sistem deteksi

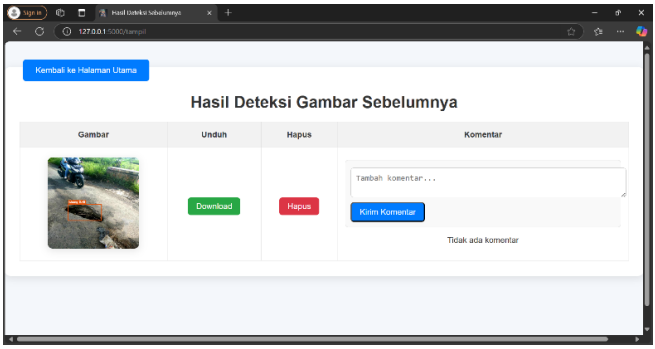
Setelah diklik maka akan langsung mengarahkan pada file index.html yang telah dirun pada web server flask. Dibawah ini tampilan website sistem deteksi saat di run.



Gambar 4. 57 Halaman awal website (index.html)



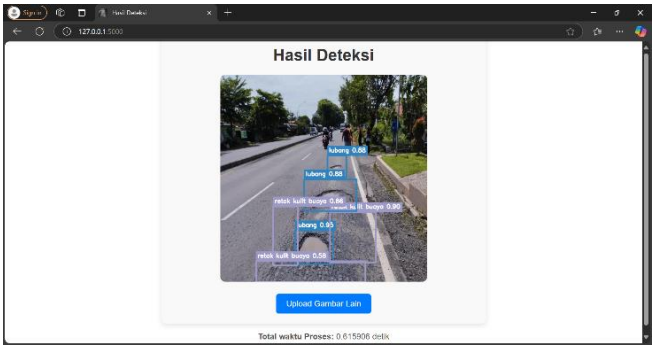
Gambar 4. 58 Tampilan halaman result.html



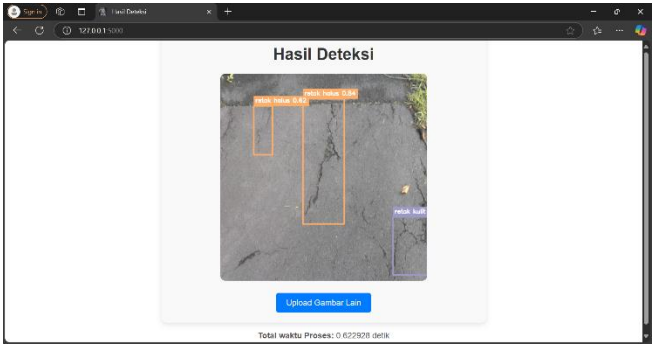
Gambar 4. 59 Tampilan halaman tampil.html

4. Pengujian Sistem Deteksi

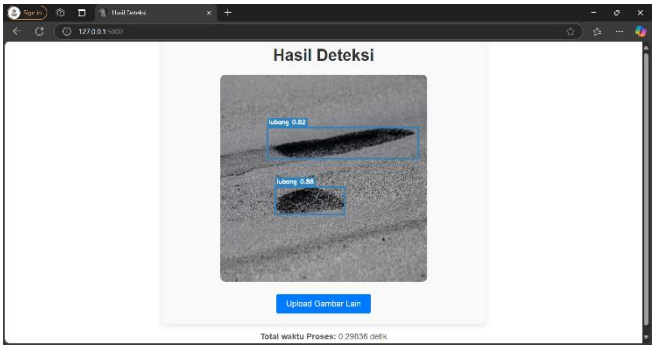
Tahap terakhir pada penelitian ini yaitu melakukan uji coba website deteksi yang telah diberi implementasi model YOLOv5. Uji coba dilakukan dengan menginputkan citra gambar dan menghitung waktu proses gambar pada halaman website. Untuk sekenario pengujian ini akan dilakukan dengan menginputkan 5 gambar yang berbeda.



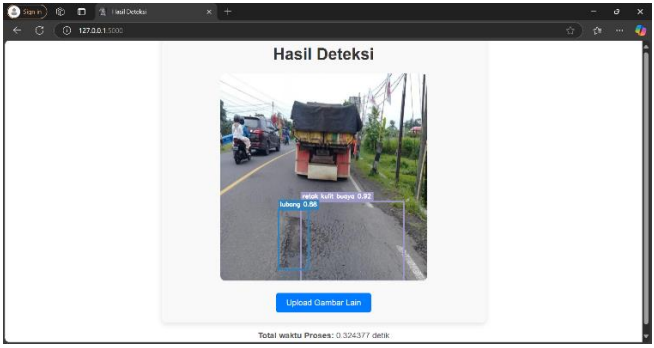
Gambar 4. 60 Uji coba webiste Input gambar 1



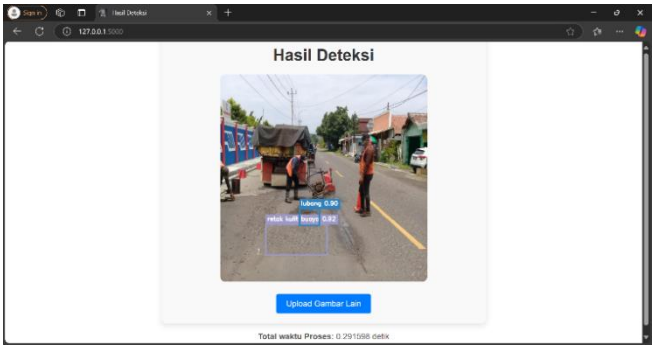
Gambar 4. 61 Uji coba webiste Input gambar 2



Gambar 4. 62 Uji coba webiste Input gambar 3



Gambar 4. 63 Uji coba webiste Input gambar 4



Gambar 4. 64 Uji coba webiste Input gambar 5

Tabel 4. 2 Tabel hasil uji coba website sistem deteksi

No	Gambar	Kelas yang terdeteksi	Waktu
1	Input 1	Lubang : 3 Retak kulit buaya : 3	0.615906 detik
2	Input 2	Retak halus : 2 Retak kulit buaya : 1	0.622928 detik
3	Input 3	Lubang : 2	0.29836 detik
4	Input 4	Lubang : 1 Retak kulit buaya : 1	0.324377 detik
5	Input 5	Lubang : 1 Retak kulit buaya : 1	0.291598 detik
Rata-rata			0.430634 detik

Berdasarkan gambar dan tabel diatas dapat dilihat bahwa website sistem deteksi dapat mendeteksi kerusakan jalan dengan baik dan dalam waktu yang cukup singkat yaitu rata-rata di 0.43 detik.

BAB V
KESIMPULAN

A. Kesimpulan

Berdasarkan hasil penelitian mengenai Implementasi Model Deep Learning dalam Deteksi dan Klasifikasi Jenis Kerusakan Jalan Aspal dengan Algoritma You Only Look Once (YOLO)v5 maka diperoleh hasil sebagai berikut :

1. Algoritma YOLOv5 terbukti efektif dalam mendeteksi dan mengklasifikasikan 3 jenis kerusakan jalan yang diteliti. Kerusakan jalan lubang dapat dideteksi dengan nilai confidence tertinggi di angka 0.94, retak kulit buaya memiliki nilai confidence tertinggi di angka 0.92 dan retak halus memiliki nilai confidence 0.84.
2. Berdasarkan hasil pengujian yang dilakukan, algoritma YOLOv5 menunjukan kinerja yang baik dalam mendeteksi dan mengklasifikasikan jenis kerusakan jalan aspal. Hal ini dibuktikan dengan perolehan nilai mAP (*mean Average Precison*) sebesar 0.797 pada saat pengujian. Nilai mAP ini menunjukan bahwa algoritma YOLOv5 mampu mendeteksi kerusakan jalan berupa retak halus, retak kulit buaya dan lubang dengan presisi yang baik.

3. Integrasi algoritma YOLOv5 dengan framework flask untuk membangun sistem deteksi berbasis web menunjukan hasil yang cukup memuaskan. Dengan pengujian menggunakan input 5 gambar, website dapat memproses gambar dalam waktu kurang dari satu detik dengan rata-rata waktu deteksi selama 0.43 detik, yang menunjukan bahwa sistem deteksi dapat memberikan hasil deteksi dengan sangat cepat. Hal ini membuktikan bahwa integrasi YOLOv5 dengan Flask dapat menghasilkan sistem deteksi yang efisien dalam memproses dan menampilkan hasil deteksi secara langsung di halaman web.

B. Saran

Berdasarkan hasil penelitian yang sudah ditampilkan sebelumnya. Peneliti menyadari bahwa pada penelitian ini terdapat beberapa kekurangan dan jauh dari kata sempurna. Maka dari itu saran dari peneliti untuk pengembangan kedepannya adalah sebagai berikut.

1. Pada bagian kelas kerusakan dapat ditambah lagi untuk jenis kerusakan jalan lain. Selain itu dataset juga bisa lebih ditingkatkan sehingga kemampuan model dalam mendeteksi kerusakan jalan akan semakin akurat.

2. Untuk website bisa lebih dikembangkan baik dari segi fitur ataupun desain agar pengguna dapat melakukan deteksi kerusakan jalan secara lebih baik dan lebih optimal.

DAFTAR PUSTAKA

Alexandrova, S., Tatlock, Z., & Cakmak, M. (2015). Roboflow: A Flow-Based Visual Programming Language For Mobile Manipulation Tasks. 2015 Ieee International Conference On Robotics And Automation (Icra), 5537–5544. Https://Doi.Org/10.1109/Icra.2015.7139973

Amwin, A. (2021). Deteksi Dan Klasifikasi Kendaraan Berbasis Algoritma You Only Look Once (Yolo). Universitas Islam Indonesia.

Andono, P. N., & Sutojo, T. (2018). Pengolahan Citra Digital. Penerbit Andi.

Ardiansyah, M. R., Supit, Y., & Said, M. S. (2022). Sistem Visi Komputer Untuk Kalkulasi Kepadatan Kendaraan Menggunakan Algoritma Yolo. Simtek : Jurnal Sistem Informasi Dan Teknik Komputer, 7(1), 52–59. Https://Doi.Org/10.51876/Simtek.V7i1.123

Cahyani, P. A. (2023). Sistem Perhitungan Kendaraan Menggunakan Algoritma Yolov5 Dan Deepsort.

Christlein, V., Spranger, L., Seuret, M., Nicolaou, A., Kral, P., & Maier, A. (2019). Deep Generalized Max Pooling. 2019 International Conference On Document Analysis And Recognition (Icdar), 1090–1096. Https://Doi.Org/10.1109/Icdar.2019.00177

Danial, N. H., & Setiawati, D. (2024). Convolutional Neural Network (Cnn) Based On Artificial Intelligence In Periodontal Diseases Diagnosis. Interdental Jurnal Kedokteran Gigi (Ijkg), 20(1), 139–148. Https://Doi.Org/10.46862/Interdental.V20i1.8641

Du, L., Zhang, R., & Wang, X. (2020). Overview Of Two-Stage Object Detection Algorithms. Journal Of Physics: Conference Series, 1544(1), 012033. Https://Doi.Org/10.1088/1742-6596/1544/1/012033

Eka Putra, W. S. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (Cnn) Pada Caltech 101. Jurnal Teknik Its, 5(1). Https://Doi.Org/10.12962/J23373539.V5i1.15696

Fang, J., Liu, Q., & Li, J. (2021). A Deployment Scheme Of Yolov5 With Inference Optimizations Based On The Triton Inference Server. 2021 Ieee 6th International Conference On Cloud Computing And Big Data Analytics (Icccbda), 441–445. Https://Doi.Org/10.1109/Icccbda51879.2021.9442557

Fetzer, J. H. (1990). What Is Artificial Intelligence? (Pp. 3–27). Https://Doi.Org/10.1007/978-94-009-1900-6_1

Gunawan, K. I., & Santoso, J. (2021). Multilabel Text Classification Menggunakan Svm Dan Doc2vec Classification Pada Dokumen Berita Bahasa Indonesia. Journal Of Information System,Graphics, Hospitality And Technology, 3(01), 29–38. Https://Doi.Org/10.37823/Insight.V3i01.126

Hamadi, A. M. (2019). Autonomous Quadrotor Control Using Convolutional Neural Autonomous Quadrotor Control Using Convolutional Neural Networks Networks. Https://Repository.Rit.Edu/Theses

Harani, N. H., Prianto, C., & Hasanah, M. (2019). Deteksi Objek Dan Pengenalan Karakter Plat Nomor Kendaraan Indonesia Menggunakan Metode Convolutional Neural Network (Cnn) Berbasis Python. Jurnal Teknik Informatika, 11, 47–53.

Hidayat, R. (2023). Implementasi Algoritma You Only Look Once (Yolo) V5 Untuk Klasifikasi Jenis Monyet.

Hidayatulloh, M. S. (2021). Sistem Pengenalan Wajah Menggunakan Metode Yolo (You Only Look Once)}. Universitas Dinamika.

Istri Lestari, I. G. A., Angga Diputera, I. G., Kubon Tubuh, I. K. D., & Jiman, A. S. (2022). Analisis Penyebab Dan Dampaknya Kerusakan Infrastruktur Jalan Terhadap Para Pengguna Jalan Dan Masyarakat Sekitar. Jurnal Ilmiah Kurva Teknik, 11(2), 32–36. Https://Doi.Org/10.36733/Jikt.V11i2.5427

Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A Review Of Yolo Algorithm Developments. Procedia Computer

Science, 199, 1066–1073. Https://Doi.Org/10.1016/J.Procs.2022.01.135
Kemajou, V. N., Bao, A., & Germain, O. (2019, April 26). Wellbore Schematics To Structured Data Using Artificial Intelligence Tools. Day 1 Mon, May 06, 2019. Https://Doi.Org/10.4043/29490-Ms
Kuroki, M. (2021). Using Python And Google Colab To Teach Undergraduate Microeconomic Theory. International Review Of Economics Education, 38, 100225. Https://Doi.Org/10.1016/J.Iree.2021.100225
Leriansyah, M., & Kurniawardhani, A. (2020). Klasifikasi Dan Perhitungan Kendaraan Untuk Mengetahui Arus Kepadatan Lalu Lintas Menggunakan Metode Yolo. Automata, 1.
Marr, D. (2010). Vision: A Computational Investigation Into The Human Representation And Processing Of Visual Information. Mit Press.
Mufid, M. R., Basofi, A., Al Rasyid, M. U. H., Rochimansyah, I. F., & Rokhim, A. (2019). Design An Mvc Model Using Python For Flask Framework Development. 2019 International Electronics Symposium (Ies), 214–219. Https://Doi.Org/10.1109/Elecsym.2019.8901656
Peryanto, A., Yudhana, A., & Umar, R. (2020). Klasifikasi Citra Menggunakan Convolutional Neural Network Dan K Fold Cross Validation. Journal Of Applied Informatics And Computing, 4(1), 45–51. Https://Doi.Org/10.30871/Jaic.V4i1.2017
Pramestya, R. H. (2018). Deteksi Dan Klasifikasi Kerusakan Jalan Aspal Menggunakan Metode Yolo Berbasis Citra Digital.
Pratiwi, H. A., Cahyanti, M., & Lamsani, M. (2021). Implementasi Deep Learning Flower Scanner Menggunakan Metode Convolutional Neural Network. Sebatik, 25(1), 124–130. Https://Doi.Org/10.46984/Sebatik.V25i1.1297
Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016, June). You Only Look Once: Unified, Real-Time Object Detection.

Proceedings Of The Ieee Conference On Computer Vision And Pattern Recognition (Cvpr).

Sandipan, D. (2018). Hands-On Image Processing With Python: Expert Techniques For Advanced Image Analysis And Effective Interpretation Of Image Data. Packt Publishing Ltd.

Sharma, N., Sharma, R., & Jindal, N. (2021). Machine Learning And Deep Learning Applications-A Vision. Global Transitions Proceedings, 2(1), 24–28. Https://Doi.Org/10.1016/J.Gltp.2021.01.004

Soviany, P., & Ionescu, R. T. (2018). Optimizing The Trade-Off Between Single-Stage And Two-Stage Deep Object Detectors Using Image Difficulty Prediction. 2018 20th International Symposium On Symbolic And Numeric Algorithms For Scientific Computing (Synasc), 209–214. Https://Doi.Org/10.1109/Synasc.2018.00041

Suhardin, I., Patombongi, A., & Islah, A. M. (2021). Mengidentifikasi Jenis Tanaman Berdasarkan Citra Daun Menggunakan Algoritma Convolutional Neural Network. Simtek : Jurnal Sistem Informasi Dan Teknik Komputer, 6(2), 100–108. Https://Doi.Org/10.51876/Simtek.V6i2.101

Sultoni, M. I., Hidayat, B., & Subandrio, A. S. (2019). Klasifikasi Jenis Batuan Beku Melalui Citra Berwarna Dengan Menggunakan Metode Local Binary Pattern Dan K-Nearest Neighbor. Tektrika-Jurnal Penelitian Dan Pengembangan Telekomunikasi, Kendali, Komputer, Elektrik, Dan Elektronika, 4, 10–15.

Suryadharma, H., & Susanto, B. (1999). Rekayasa Jalan Raya. In Universitas Atma Jaya. Universitas Atma Jaya.

Suryowinoto, A., & Hamid, A. (2017). Penggunaan Pengolahan Citra Digital Dengan Algoritma Edge Detection Dalam Mengidentifikasi Kerusakan Kontur Jalan. Emin. Nas. Sains Dan Teknol. Terap. V, 149–154.

Szeliski, R. (2022). Computer Vision: Algorithms And Applications. Springer Nature.

Yudaningrum, F., & Ikhwanudin, I. (2017). Identifikasi Jenis Kerusakan Jalan (Studi Kasus Ruas Jalan Kedungmundu-Meteseh). Teknika, 12(2). Https://Doi.Org/10.26623/Teknika.V12i2.638

Zhao, Z.-Q., Zheng, P., Xu, S.-T., & Wu, X. (2019). Object Detection With Deep Learning: A Review. Ieee Transactions On Neural Networks And Learning Systems, 30(11), 3212–3232. Https://Doi.Org/10.1109/Tnnls.2018.2876865

Zhou, F., Zhao, H., & Nie, Z. (2021). Safety Helmet Detection Based On Yolov5. 2021 Ieee International Conference On Power Electronics, Computer Applications (Icpeca), 6–11. Https://Doi.Org/10.1109/Icpeca51329.2021.9362711

Zou, Z., Chen, K., Shi, Z., Guo, Y., & Ye, J. (2023). Object Detection In 20 Years: A Survey. Proceedings Of The Ieee, 111(3), 257–276. Https://Doi.Org/10.1109/Jproc.2023.3238524

LAMPIRAN

Lampiran 1 : Lembar Pengesahan Proposal

LEMBAR PENGESAHAN

Judul : Implementasi *Model Deep Learning* dalam Deteksi dan Klasifikasi Jenis kerusakan Jalan Aspal dengan Algoritma *You Only Look Once* (YOLO)v5

Nama : Mochammad Ali Ridho Fathoni

Nim : 2008096061

Jurusan : Teknologi Informasi

Telah diujikan dalam sidang komprehensif oleh Dewan Penguji Fakultas Sains dan Teknologi UIN Walisongo Semarang dan dapat diterima sebagai salah satu syarat memperoleh gelar sarjana dalam program studi Teknologi Informasi.

Penguji I	Penguji II
Dr. Khotibul Umam, S.T.,M.Kom	Dr. Masy Ari Ulinuha, S.T., M.T
NIP. 19790827 201101 1 007	NIP. 1981081 2201101 1 007

Penguji III	Penguji IV
Siti Nur'aini, M. Kom	Mokhamad Iklil Mustofa, M.Kom.
NIP. 19840131 201801 2 001	NIP. 19880807 201903 1 010

Lampiran 2 : Source Code detekapp.py

```
import io
import os
import json
import datetime
from flask import Flask, render_template, request, redirect,
url_for
from PIL import Image
import cv2
import torch
import numpy as np
from pathlib import Path
from yolov5.models.common import DetectMultiBackend
from yolov5.utils.general import non_max_suppression,
scale_boxes
from yolov5.utils.torch_utils import select_device
import matplotlib.pyplot as plt # Import matplotlib untuk
warna

# Patch untuk PosixPath di Windows
import pathlib
if os.name == 'nt': # Jika di Windows
 temp = pathlib.PosixPath
 pathlib.PosixPath = pathlib.WindowsPath
```

```
# Inisialisasi Flask
app = Flask(__name__)
app.config['UPLOAD_FOLDER'] = 'static'
DATETIME_FORMAT = "%Y-%m-%d_%H-%M-%S-%f"

# Load model YOLOv5
MODEL_PATH = "models/best.pt"
device = select_device('CPU') # Pilih perangkat (GPU jika
tersedia, atau CPU)
model = DetectMultiBackend(MODEL_PATH,
device=device)
names = model.names
img_size = 640 # Resolusi input gambar

# Fungsi untuk menghasilkan warna unik berdasarkan
kelas
# Fungsi untuk menghasilkan warna unik berdasarkan
kelas
# Fungsi untuk menghasilkan warna unik berdasarkan
kelas
def generate_colors(num_colors):
 # Menggunakan colormap dari matplotlib untuk
mendapatkan warna unik
```

```
 colormap = plt.get_cmap("tab20c") # Menggunakan
colormap yang memiliki berbagai warna kontras
 colors = [tuple(int(x * 255) for x in colormap(i /
num_colors)[:3]) for i in range(num_colors)]
 return colors

# Mengambil warna yang sesuai dengan jumlah kelas
colors = generate_colors(len(names))

# Fungsi untuk menggambar bounding box dengan label
yang lebih jelas
def draw_boxes(image, boxes, names, colors):
 """
 Menggambar bounding box dengan label yang sesuai
panjang teks.
 """
 for *box, conf, cls in boxes:
  cls = int(cls) # Pastikan kelas adalah integer
  x1, y1, x2, y2 = map(int, box)
  label = f"{names[cls]} {conf:.2f}"
  color = colors[cls] # Warna unik untuk setiap kelas

  # Gambar bounding box dengan ketebalan 3
  cv2.rectangle(image, (x1, y1), (x2, y2), color, 3)
```

```
  # Ukuran teks label
  font_scale = 0.6
  font_thickness = 2
  (label_width, label_height), baseline =
cv2.getTextSize(label, cv2.FONT_HERSHEY_SIMPLEX,
font_scale, font_thickness)

  # Padding untuk latar belakang label
  padding = 5
  label_x2 = x1 + label_width + 2 * padding
  label_y1 = max(0, y1 - label_height - baseline - 2 * padding)
# Pastikan tidak keluar dari gambar
  label_y2 = y1

  # Gambar latar belakang label
  cv2.rectangle(image, (x1, label_y1), (label_x2, label_y2),
color, -1)

  # Tambahkan label teks di atas latar belakang
  text_x = x1 + padding
  text_y = y1 - padding - baseline
```

```
  cv2.putText(image, label, (text_x, text_y),
cv2.FONT_HERSHEY_SIMPLEX, font_scale, (255, 255, 255),
font_thickness)

 return image

# Mengambil nama file gambar yang ada
def get_image_filenames():
 image_dir = pathlib.Path(app.config['UPLOAD_FOLDER'])
 return [str(file.name) for file in image_dir.glob('*.png')] #
Menyesuaikan dengan ekstensi file yang digunakan

# Mengambil komentar dari file JSON
def get_comments():
 if os.path.exists('comments.json'):
  with open('comments.json', 'r') as f:
   return json.load(f)
 return {}

# Menyimpan komentar di file JSON
def save_comments(comments):
 with open('comments.json', 'w') as f:
  json.dump(comments, f, indent=4)
```

```
@app.route("/", methods=["GET", "POST"])
def index():
 if request.method == "POST":
  # Catat waktu awal pengunggahan
  start_time = datetime.datetime.now()

  # Periksa apakah file diunggah
  if "file" not in request.files:
   return redirect(request.url)
  file = request.files["file"]
  if not file:
   return redirect(request.url)

  # Catat waktu pengunggahan
  upload_time = start_time.strftime(DATETIME_FORMAT)
  print(f"File uploaded at: {upload_time}") # Opsional: Log
waktu upload ke console

  # Baca gambar yang diunggah
  img_bytes = file.read()
  img = Image.open(io.BytesIO(img_bytes)).convert('RGB')
  original_size = img.size

  # Lakukan preprocessing
```

```
  img_resized = img.resize((img_size, img_size))
  img_array = np.array(img_resized)
  img_tensor =
torch.from_numpy(img_array).float().permute(2, 0,
1).unsqueeze(0).to(device) / 255.0

  # Lakukan deteksi
  pred = model(img_tensor)
  pred = non_max_suppression(pred)[0] # NMS untuk
menghapus prediksi duplikat

  # Gambar bounding box pada gambar asli
  if pred is not None and len(pred):
   pred[:, :4] = scale_boxes(img_tensor.shape[2:], pred[:, :4],
img_array.shape[:2]).round()
   img_array = draw_boxes(np.array(img), pred, names,
colors)

  # Simpan hasil gambar
  now_time =
datetime.datetime.now().strftime(DATETIME_FORMAT)
  output_path = os.path.join(app.config['UPLOAD_FOLDER'],
f"{now_time}.png")
  Image.fromarray(img_array).save(output_path)
```

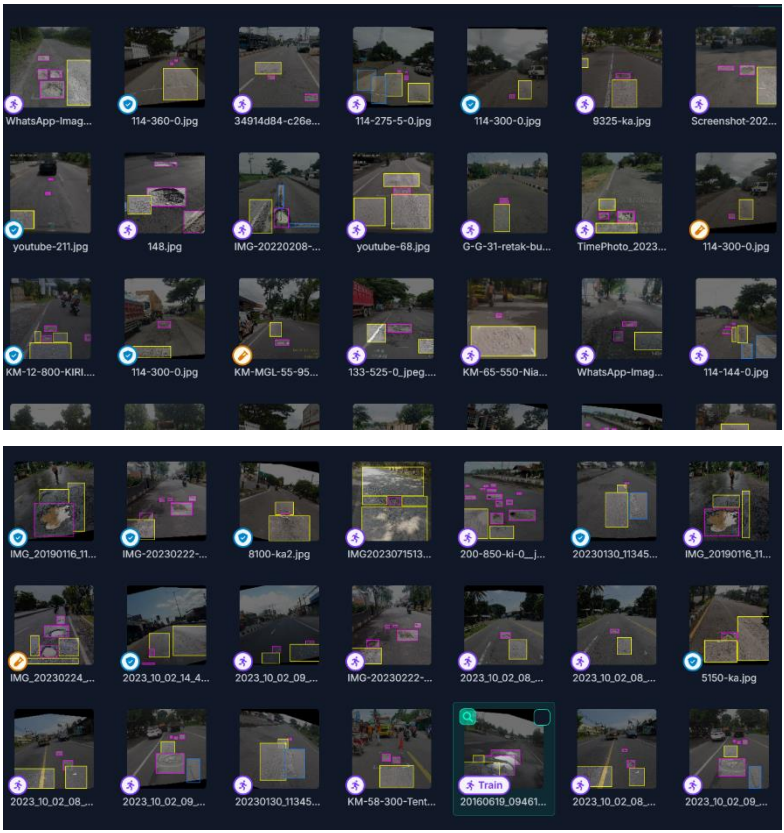
```
  # Catat waktu proses selesai
  end_time = datetime.datetime.now()
  process_time = end_time.strftime(DATETIME_FORMAT)
  print(f"File processed at: {process_time}") # Opsional: Log
waktu proses ke console

  # Hitung durasi total
  total_duration = (end_time - start_time).total_seconds()
  print(f"Total processing time: {total_duration} seconds") #
Opsional: Log durasi ke console

  # Render halaman hasil dengan waktu upload, waktu
proses, dan durasi total
  return render_template(
   "result.html",
   result_image_url=url_for('static',
filename=f"{now_time}.png"),
   upload_time=upload_time,
   process_time=process_time,
   total_duration=total_duration
  )
 return render_template("index.html")
```

```
@app.route("/tampil", methods=["GET"])
def tampil():
 images = get_image_filenames()
 comments = get_comments()
 return render_template("tampil.html", images=images,
comments=comments)

@app.route("/delete/<filename>", methods=["POST"])
def delete_image(filename):
 image_path = os.path.join(app.config['UPLOAD_FOLDER'],
filename)

 if os.path.exists(image_path):
  os.remove(image_path) # Hapus gambar dari folder static

 comments = get_comments()
 if filename in comments:
  del comments[filename] # Hapus komentar terkait
gambar

 save_comments(comments) # Simpan perubahan
komentar
```

```
 return redirect(url_for('tampil')) # Arahkan kembali ke
halaman tampil setelah gambar dihapus

@app.route("/add_comment/<filename>",
methods=["POST"])
def add_comment(filename):
 comment = request.form.get('comment')
 if comment:
  comments = get_comments()
  if filename not in comments:
   comments[filename] = []
  comments[filename].append(comment)
  save_comments(comments) # Simpan komentar ke dalam
file JSON

 return redirect(url_for('tampil')) # Arahkan kembali ke
halaman tampil setelah komentar ditambahkan

@app.route("/delete_comment/<filename>/<int:comment_
index>", methods=["POST"])
def delete_comment(filename, comment_index):
 # Ambil komentar yang sudah ada
 comments = get_comments()
```

```
 # Cek apakah gambar memiliki komentar
 if filename in comments:
  # Hapus komentar berdasarkan index yang diberikan
  if 0 <= comment_index < len(comments[filename]):
   comments[filename].pop(comment_index)
   save_comments(comments) # Simpan perubahan
komentar ke file JSON

 return redirect(url_for('tampil')) # Arahkan kembali ke
halaman tampil setelah komentar dihapus

if __name__ == "__main__":
 app.run(debug=True)
```

Lampiran 3 : Dataset



https://universe.roboflow.com/skripsi-hcwrz/deteksi-kerusakan-jalan-ckrvi

Lampiran 4 : Daftar Riwayat Hidup

DAFTAR RIWAYAT HIDUP

A. Identitas Diri

Nama	:	Mochammad Ali Ridho Fathoni
Tempat, Tanggal Lahir	:	Semarang, 14 Desember 2001
Alamat	:	Dempel Lor RT.01/RW.14 Muktiharjo Kidul Pedurungan Kota Semarang
HP	:	083836489507
Email	:	fathonialiridho@gmail.com

B. Riwayat Pendidikan

1. SDN Kaligawe Kota Semarang
2. MTs Tajul Ulum Kabupaten Grobogan
3. MAN 2 Kota Semarang