

**SISTEM DETEKSI KENDARAAN MENGGUNAKAN  
METODE *YOU ONLY LOOK ONCE* (YOLO) PADA  
JALANAN KOTA MEDAN**

**SKRIPSI**

**FARADILLA HAIFA FALIYA LUBIS  
191401018**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN  
2025**

**SISTEM DETEKSI KENDARAAN MENGGUNAKAN  
METODE *YOU ONLY LOOK ONCE* (YOLO) PADA  
JALANAN KOTA MEDAN**

**SKRIPSI**

Diajukan untuk melengkapi tugas dan memenuhi syarat memperoleh ijazah  
Sarjana Ilmu Komputer

**FARADILLA HAIFA FALIYA LUBIS  
191401018**



**PROGRAM STUDI S-1 ILMU KOMPUTER  
FAKULTAS ILMU KOMPUTER DAN TEKNOLOGI INFORMASI  
UNIVERSITAS SUMATERA UTARA  
MEDAN**

**2025**

**PERSETUJUAN**

Judul : SISTEM DETEKSI KENDARAAN  
MENGUNAKAN METODE *YOU ONLY  
LOOK ONCE* (YOLO) PADA JALANAN  
KOTA MEDAN

Kategori : SKRIPSI  
Nama : FARADILLA HAIFA FALIYA LUBIS  
Nomor Induk Mahasiswa : 191401018  
Program Studi : SARJANA (S-1) ILMU KOMPUTER  
Fakultas : ILMU KOMPUTER DAN TEKNOLOGI  
INFORMASI

Telah diuji dan dinyatakan lulus di Medan, 10 Januari 2025.

Komisi Pembimbing :

Pembimbing II

Pembimbing I

Dr. Maya Silvi Lydia B.Sc., M.Sc.  
NIP. 197401272002122001

Dedy Arisandi, S.T., M.Kom.  
NIP. 197908312009121002

Diketahui/Disetujui Oleh

Ketua Program Studi S-1 Ilmu Komputer



Dr. Amalia, S.T., M.T.

NIP. 197812212014042001

## **PERNYATAAN**

### **SISTEM DETEKSI KENDARAAN MENGGUNAKAN METODE *YOU ONLY LOOK ONCE* (YOLO) PADA JALANAN KOTA MEDAN**

#### **SKRIPSI**

Saya mengakui bahwa skripsi ini adalah hasil penelitian saya sendiri, kecuali beberapa kutipan dan ringkasan yang sudah disebutkan sumbernya.

Medan, 10 Januari 2025



Faradilla Haifa Faliya Lubis

191401018



## PENGHARGAAN

Puji dan syukur penulis ucapkan kehadirat Allah SWT, karena rahmat dan kuasa – Nya, penulis dapat menyelesaikan penyusunan skripsi ini sebagai syarat untuk memperoleh gelar Sarjana Komputer, pada Program Studi S1 Ilmu Komputer Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara.

Skripsi ini tentu diiringi oleh bantuan dan kehadiran dari berbagai pihak selama masa penyelesaiannya. Untuk itu, penulis ingin menyampaikan ungkapan hormat dan terima kasih yang sebesar-besarnya atas bimbingan, do'a, dukungan, ilmu, dan pelajaran dari semua pihak selama penyusunan skripsi ini, diantaranya:

1. Bapak Dr. Muryanto Amin, S.Sos., M.Si. selaku Rektor Universitas Sumatera Utara.
2. Ibu Dr. Maya Silvi Lidya, B.Sc., M.Sc. selaku Dekan Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara dan selaku dosen pembimbing II yang telah memberikan bimbingan, kritik, motivasi, dan saran kepada penulis dalam menyelesaikan skripsi ini.
3. Ibu Dr. Amalia, S.T, M.T. selaku Ketua Program Studi S-1 Ilmu Komputer Universitas Sumatera Utara.
4. Bapak Dedy Arisandi, S.T., M.Kom. selaku Dosen Pembimbing I yang telah memberikan ilmu dan bimbingan dalam penyusunan skripsi ini.
5. Ibu Sri Melvani Hardi, S.Kom., M.Kom. selaku Dosen Penasihat Akademik selama penulis menempuh pendidikan di Program Studi S-1 Ilmu Komputer.
6. Seluruh staf pengajar dan pegawai Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Sumatera Utara yang telah membantu dalam proses penyusunan skripsi ini.
7. Kedua orang tua, Ayahanda Fahmi Lubis dan Ibunda Emeliya yang senantiasa memberikan do'a, cinta, kasih sayang, dukungan moril dan materil, motivasi, dan semangatnya kepada saya sebagai penulis hingga penulis dapat menyelesaikan tugas akhir ini.

8. Adik tercinta, Haurra Effarissa Lubis dan M. Fairuz Falifasya Lubis yang senantiasa mengiringi penulis dengan do'a dan semangat.
9. Bunde tercinta, Salbiah Lubis dan Amangboru Adil Solihin Putera yang mendoakan dan mendukung penulis sehingga mempermudah penulis dalam melakukan penyusunan tugas akhir.
10. Sahabat seperjuangan, Farika Aini Nasution, Cindy M.T. Siregar, Iftitah Maghfirah Kesuma Putri, dan M. Amirul Ilmi.
11. Seluruh teman-teman Kom C dan Stambuk 2019 Program Studi S-1 Ilmu Komputer yang telah memberikan semangat kepada penulis.
12. Keluarga IMILKOM USU, Badan Pengurus Harian, dan teman-teman Presidium Periode 2022/2023 yang telah memberikan banyak ilmu dan pengalaman kepada penulis selama masa perkuliahan.
13. Dan seluruh pihak yang kehadirannya telah membantu penulis secara langsung maupun tidak langsung yang tidak dapat disebutkan satu per satu.

Semoga Allah SWT melimpahkan berkah kepada semua pihak yang telah memberikan bantuan, semangat, perhatian, serta dukungan kepada penulis dalam menyelesaikan skripsi ini. Penulis berharap besar atas kebermanfaatan skripsi ini bagi seluruh pihak yang membutuhkan.

Medan, 10 Januari 2025

Penulis,



Faradilla Haifa Faliya Lubis

191401018



## ABSTRAK

Pemantauan terhadap kondisi lalu lintas sangatlah penting dengan bertambahnya jumlah penduduk maka akan mempengaruhi banyaknya kendaraan di Kota Medan sebagai upaya untuk pengaturan dan kelancaran lalu lintas. Kota Medan merupakan salah satu daerah dengan tingkat kepadatan kendaraan yang cukup tinggi, terutama di kawasan lalu lintas utama. Beragam jenis kendaraan melintas setiap hari, sehingga diperlukan teknologi yang mampu mendeteksi dan mengidentifikasi kendaraan secara akurat dan efisien. Dalam penelitian ini, penulis menggunakan metode YOLO (*You Only Look Once*) sebagai algoritma deteksi objek untuk menguji kemampuannya dalam mengenali berbagai jenis kendaraan yang melintas di lalu lintas Kota Medan. Penelitian ini menggunakan objek berupa kendaraan mobil, sepeda motor, bus, truck, dan becak. Sistem ini dibangun dengan menggunakan metode *You Only Look Once* yang melakukan pengelompokkan jenis kendaraan dan menghitung kendaraan. Dari hasil pengujian ini berdasarkan metode YOLO menggunakan 4 video uji dengan durasi video diatas 40 detik terdapat 180 kendaraan telah berhasil terdeteksi dengan benar dari total 189 kendaraan. Dari perhitungan akurasi dan perhitungan performa lainnya, didapati bahwa sistem yang mengimplementasikan metode *You Only Look Once* (YOLO) v5 ini memiliki akurasi sebesar 95%.

**Kata Kunci:** Sistem Deteksi Kendaraan, Deteksi Objek, Metode *You Only Look Once*

**VEHICLE DETECTION SYSTEM USING THE YOU ONLY LOOK ONCE  
(YOLO) METHOD IN MEDAN CITY STREET**

**ABSTRACT**

*Monitoring traffic conditions is very important with the increasing population, it will affect the number of vehicles in Medan City as an effort to regulate and smooth traffic. Medan City is one of the areas with a fairly high level of vehicle density, especially in the main traffic area. Various types of vehicles pass every day, so technology is needed that can detect and identify vehicles accurately and efficiently. In this study, the author uses the YOLO (You Only Look Once) method as an object detection algorithm to test its ability to recognize various types of vehicles passing through Medan City traffic. This study uses objects in the form of cars, motorcycles, buses, trucks, and pedicabs. This system is built using the You Only Look Once method which groups vehicle types and counts vehicles. From the results of this test based on the YOLO method using 4 test videos with a video duration of over 40 seconds, 180 vehicles have been successfully detected correctly out of a total of 189 vehicles. From the calculation of accuracy and other performance calculations, it was found that the system that implements the You Only Look Once (YOLO) v5 method has an accuracy of 95%.*

**Keyword:** *Vehicle Detection System, Object Detection, You Only Look Once Method.*



## DAFTAR ISI

PERSETUJUAN .....	iii
PERNYATAAN .....	iv
PENGHARGAAN.....	v
ABSTRAK .....	vii
<i>ABSTRACT</i> .....	viii
DAFTAR ISI .....	ix
DAFTAR GAMBAR.....	xi
DAFTAR TABEL.....	xii
DAFTAR LAMPIRAN .....	xiii
<b>BAB 1 PENDAHULUAN.....</b>	<b>1</b>
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	2
1.3. Batasan Masalah.....	3
1.4. Tujuan Penelitian.....	3
1.5. Manfaat Penelitian.....	3
1.6. Metodologi Penelitian .....	4
1.7. Sistematika Penulisan.....	5
<b>BAB 2 LANDASAN TEORI.....</b>	<b>6</b>
2.1. Kendaraan.....	6
2.2. Deteksi Objek.....	6
2.3. Metode <i>You Only Look Once</i> (YOLO).....	9
2.4. You Only Look Once (YOLO) v5.....	12
2.4.1. <i>Backbone</i> .....	13
2.4.2. <i>Neck</i> .....	13
2.4.3. <i>Head</i> .....	13
2.5. Penelitian Relevan.....	15
<b>BAB 3 ANALISIS DAN PERANCANGAN SISTEM.....</b>	<b>16</b>
3.1. Data yang digunakan .....	16
3.2. Arsitektur Umum.....	16

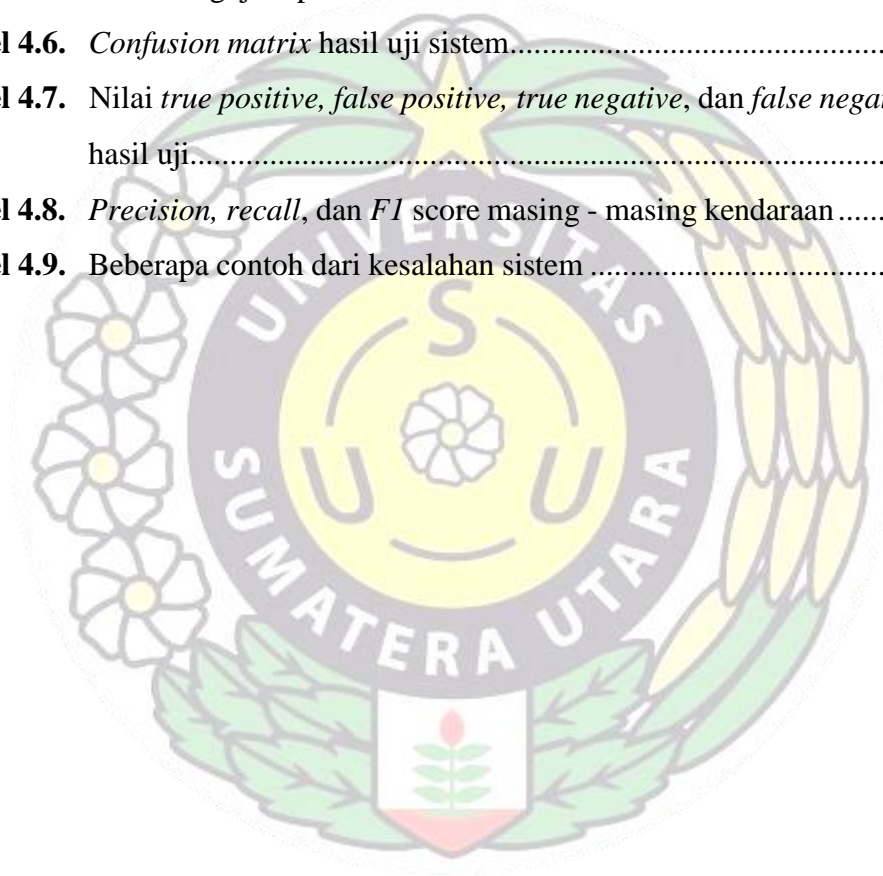
3.3. Pemodelan Sistem .....	18
3.4. <i>Sistem Pre-Processing Data</i> .....	19
3.4.1. Akuisisi Data .....	19
3.4.2. <i>Pre-processing Data</i> .....	20
3.5. <i>Flowchart</i> .....	22
<b>BAB 4 IMPLEMENTASI DAN PENGUJIAN</b> .....	24
4.1. Kebutuhan Sistem .....	24
4.1.1. Perangkat Lunak .....	24
4.1.2. Perangkat Keras .....	24
4.2. Hasil Akuisisi Data .....	24
4.3. Training Dan Validasi .....	25
4.4. Hasil Pengujian Algoritma YOLO Menggunakan Google Colab .....	28
4.5. Hasil Pengujian Penghitung Kendaraan yang Melintas .....	33
4.6. Pembahasan .....	43
<b>BAB 5 KESIMPULAN DAN SARAN</b> .....	44
5.1. Kesimpulan .....	44
5.2. Saran .....	44
<b>DAFTAR PUSTAKA</b> .....	45
<b>LAMPIRAN</b> .....	47

## DAFTAR GAMBAR

<b>Gambar 2.1.</b>	Deteksi objek pada kendaraan <i>autonomous</i> .....	7
<b>Gambar 2.2.</b>	Plot skema (a) deteksi objek “ <i>One-stage Detector</i> ” dan (b) deteksi objek “ <i>Two-stage Detector</i> ” .....	8
<b>Gambar 2.3.</b>	<i>Road map</i> perkembangan deteksi objek selama dua dekade terakhir .....	8
<b>Gambar 2.4.</b>	Konsep pendeteksian algoritma YOLO .....	9
<b>Gambar 2.5.</b>	Prediksi <i>bounding box</i> yang beragam .....	10
<b>Gambar 2.6.</b>	Pembuangan <i>bounding box</i> yang nilainya kurang dari <i>threshold</i> yang ditentukan .....	10
<b>Gambar 2.7.</b>	Arsitektur YOLO.....	11
<b>Gambar 2.8.</b>	Arsitektur YOLOv5.....	12
<b>Gambar 2.9.</b>	Visualisasi <i>Loss Function</i> pada YOLOv5 .....	14
<b>Gambar 3.1.</b>	Arsitektur Umum Penelitian.....	18
<b>Gambar 3.2.</b>	Blok Diagram Aplikasi Penghitung Kendaraan .....	18
<b>Gambar 3.3.</b>	Diagram blok sistem <i>processing</i> data.....	19
<b>Gambar 3.4.</b>	Tampilan <i>Roboflow</i> Fitur <i>Convert Video to JPG</i> .....	20
<b>Gambar 3.5.</b>	Hasil <i>Convert Video to JPG</i> sebelum dilakukan pelabelan.....	20
<b>Gambar 3.6.</b>	Pemberian label pada objek .....	21
<b>Gambar 3.7.</b>	Beberapa datasheet yang telah melalui proses pelabelan.....	21
<b>Gambar 3.8.</b>	Flowchart Sistem .....	22
<b>Gambar 4.1.</b>	Plot <i>Box</i> , <i>Objectness</i> , <i>Classification</i> , <i>Precision</i> , <i>Recall</i> setiap periode untuk <i>Training</i> dan Validasi Dataset .....	27
<b>Gambar 4.2.</b>	Hasil <i>training</i> menggunakan model YOLOv5.....	28
<b>Gambar 4.3.</b>	Confusion matrix .....	29
<b>Gambar 4.4.</b>	Kurva nilai F1 terhadap nilai <i>Confidence</i> .....	30
<b>Gambar 4.5.</b>	Kurva nilai <i>Precision</i> terhadap nilai <i>Confidence</i> .....	31
<b>Gambar 4.6.</b>	Kurva nilai <i>Recall</i> terhadap nilai <i>Confidence</i> .....	32
<b>Gambar 4.7.</b>	Kurva nilai <i>Precision</i> terhadap nilai <i>Recall</i> .....	33
<b>Gambar 4.8.</b>	Hasil Hitung Kendaraan Melintas.....	33

## DAFTAR TABEL

<b>Tabel 4.1.</b> Hasil Pengujian Google Colab.....	28
<b>Tabel 4.2.</b> Hasil Pengujian pada Video 1 .....	34
<b>Tabel 4.3.</b> Hasil Pengujian pada Video 2 .....	34
<b>Tabel 4.4.</b> Hasil Pengujian pada Video 3 .....	35
<b>Tabel 4.5.</b> Hasil Pengujian pada Video 4 .....	35
<b>Tabel 4.6.</b> <i>Confusion matrix</i> hasil uji sistem.....	36
<b>Tabel 4.7.</b> Nilai <i>true positive</i> , <i>false positive</i> , <i>true negative</i> , dan <i>false negative</i> hasil uji.....	37
<b>Tabel 4.8.</b> <i>Precision</i> , <i>recall</i> , dan <i>F1 score</i> masing - masing kendaraan .....	42
<b>Tabel 4.9.</b> Beberapa contoh dari kesalahan sistem .....	43



## DAFTAR LAMPIRAN

<b>Lampiran 1</b>	<i>Curriculum Vitae</i> .....	A-1
-------------------	-------------------------------	-----





## BAB 1 PENDAHULUAN

### 1.1 Latar Belakang

Inovasi teknologi berkembang dengan sangat pesat di dunia kontemporer. *Artificial intelligence* (AI) merujuk pada perangkat lunak yang dapat meniru proses kognitif manusia dan terus mengalami peningkatan di bidang ini. Salah satu penggunaan *Artificial intelligence* (AI) saat ini adalah pembuatan penghitung kendaraan di jalan raya yang dapat mengidentifikasi merek, model, kecepatan, dan jenis kendaraan di jalan. Kondisi lalu lintas harus dipantau secara ketat karena jumlah mobil di Indonesia diperkirakan akan terpengaruh oleh populasi negara yang terus bertambah. Di setiap negara berkembang, termasuk Indonesia, masalah kemacetan lalu lintas jalan raya sudah lama menjadi perhatian utama di kota-kota besar. Tahap perencanaan, perancangan, dan pengelolaan operasi jalan bergantung pada data volume lalu lintas yang dibutuhkan untuk mengurangi kemacetan dan mengoptimalkan pemanfaatan jalan raya.

Berdasarkan penjelasan di atas, diperlukan solusi untuk memanfaatkan teknologi deteksi objek guna mengelompokkan berbagai jenis kendaraan yang terekam di Kota Medan. Deteksi objek atau yang biasa disebut dengan *object detection* merupakan pemanfaatan komputer untuk mendeteksi dan mengidentifikasi objek tertentu. Studi yang meneliti bagaimana komputer dapat mengenali objek seperti halnya mata manusia, mencakup kemampuan untuk mendeteksi objek merupakan bagian dari ilmu *computer vision*. Dalam *computer vision*, deteksi objek dapat menganalisis video maupun gambar untuk menyelesaikan sebuah permasalahan.

Identifikasi objek hanyalah satu bidang di mana teknologi ini telah membantu kemajuan penelitian kecerdasan buatan. Teknologi ini juga dapat membantu kita mengidentifikasi objek dalam gambar. Salah satu bidang visi komputer adalah deteksi objek. Studi visi komputer meneliti bagaimana komputer dapat mengenali dan menginterpretasikan sesuatu dalam gambar (Haryono et al. 2019).

Pendeteksi objek (*Object detection*) bermanfaat untuk mengenali dan mendeteksi objek pada sebuah gambar berdasarkan dari warna, bentuk dan dari dataset yang dikumpulkan (Lin et al. 2020). Ada beberapa macam untuk membuat aplikasi pendeteksian objek, salah satunya adalah memakai metode *Convolutional Neural Network* (CNN) dan metode *You Only Look Once* (YOLO) (You et al. 2019). Sistem pendeteksi metode YOLO terbukti lebih cepat dan akurat untuk mendeteksi objek pada gambar atau citra sehingga paling sesuai jika diterapkan untuk real-time pendeteksian objek pada video (Dasgupta, Bandyopadhyay, and Chatterji 2019).

YOLO merupakan jaringan untuk mendeteksi objek sedangkan YOLOv5 adalah metode versi terbaru yang dikembangkan metode YOLO (Tan et al. 2021). Tugas pendeteksian objek untuk menentukan tempat pada sebuah gambar atau citra pada objek yang hadir dan mengklasifikasikan jenis objeknya. Jadi sederhananya ada sebuah gambar atau citra menjadi inputan, kemudian buat vektor kotak pembatas dan prediksi kelas dalam outputnya (Wei, He, and Lu 2020).

Pada penelitian ini digunakan metode *You Only Look Once* (YOLO) versi 5 yang merupakan algoritma deteksi objek waktu nyata yang unggul dalam akurasi dan kecepatan waktu inferensi. YOLO menggunakan sebuah jaringan syaraf tunggal (*single neural network*) untuk melakukan pendeteksian dan pengenalan objek yang memprediksi secara langsung *bounding box* dan probabilitas kelas.

## 1.2 Rumusan Masalah

Kota Medan merupakan salah satu daerah dengan tingkat kepadatan kendaraan yang cukup tinggi, terutama di kawasan lalu lintas utama. Beragam jenis kendaraan melintas setiap hari, sehingga diperlukan teknologi yang mampu mendeteksi dan mengidentifikasi kendaraan secara akurat dan efisien. Dalam penelitian ini, penulis menggunakan metode YOLO (*You Only Look Once*) sebagai algoritma deteksi objek untuk menguji kemampuannya dalam mengenali berbagai jenis kendaraan yang melintas di lalu lintas Kota Medan.

### 1.3 Batasan Masalah

Dalam penelitian ini, peneliti menghadapi sejumlah batasan terkait penyelesaian masalah. Berikut adalah batasan-batasan yang ditetapkan dalam penelitian ini:

1. Data citra kendaraan diambil di area jalan di Kota Medan Sumatera Utara di rekam dengan *smartphone*.
2. Kendaraan yang dijadikan objek dideteksi adalah mobil, sepeda motor, bus, truck, becak.
3. Kendaraan bergerak dalam satu arah.
4. Pengambilan citra untuk dataset dilakukan pada siang hari dalam kondisi cuaca yang cerah.

### 1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk mengembangkan sistem deteksi kendaraan di jalanan Kota Medan, yang akan diimplementasikan dalam bentuk situs website dengan metode *You Only Look Once* (YOLO) versi 5.

### 1.5 Manfaat Penelitian

Adapun manfaat yang diperoleh dari penelitian ini adalah sebagai berikut.

1. Dapat memudahkan proses peningkatan kinerja lalu lintas.
2. Sebagai solusi untuk dapat mengelompokkan jenis kendaraan yang terekam di kota Medan dengan menggunakan kemampuan teknologi pendeteksi objek.
3. Dapat dijadikan sebagai referensi untuk pekerjaan lainnya yang berkaitan dengan deteksi objek kendaraan.

## 1.6 Metodologi Penelitian

Adapun tahapan metode yang digunakan dalam penelitian adalah:

1. Studi Pustaka

Pada tahap ini, penelitian dimulai dengan mengumpulkan referensi dan tinjauan pustaka dari berbagai sumber berupa buku, jurnal, makalah, artikel, dan penelitian terdahulu yang relevan dengan Sistem Deteksi Kendaraan dan metode *You Only Look Once* (YOLO) versi 5.

2. Studi Lapangan

Pada tahap ini, informasi untuk kepentingan penelitian dikumpulkan dengan mengambil beberapa video yang dibutuhkan. Video tersebut yang nantinya akan dijadikan sumber data yang akan digunakan dalam aplikasi tersebut.

3. Analisis dan Perancangan Sistem

Pada tahap ini, peneliti menganalisis segala sesuatu yang dibutuhkan sistem dalam penelitian kemudian membuat *flowchart*, *use case diagram*, *sequence diagram* dan desain *user interface* sesuai dengan kebutuhan sistem.

4. Implementasi

Pada tahap ini, aplikasi dibuat berdasarkan *flowchart* yang telah direncanakan sebelumnya. Bahasa pemrograman menggunakan *Python* dan Visual Studio Code digunakan sebagai IDE untuk membuat aplikasi dalam penelitian ini.

5. Pengujian

Pada tahap ini, sistem di uji keberhasilan dan akurasi berdasarkan data jenis dan jumlah kendaraan yang terdeteksi oleh sistem.

6. Dokumentasi

Pada tahap ini, peneliti akan melakukan dokumentasi dan penulisan laporan dari hasil penelitian mengenai aplikasi yang telah dibangun dalam format skripsi.



## 1.7 Sistematika Penulisan

Adapun sistematika penulisan yang digunakan pada skripsi ini terdiri dari beberapa bagian sebagai berikut.

### **BAB 1            PENDAHULUAN**

Bab ini berisikan tentang latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, serta sistematika penulisan yang digunakan pada skripsi ini.

### **BAB 2            LANDASAN TEORI**

Bab ini menjelaskan tentang beberapa teori mendasar terkait jenis kendaraan, deteksi objek, metode *You Only Look Once* (YOLO) versi 5.

### **BAB 3            ANALISIS DAN PERANCANGAN SISTEM**

Pada bab ini, dirincikan mengenai analisis kebutuhan dan analisis proses dari sistem beserta perancangan alur dan aplikasi sistem deteksi jenis dan jumlah kendaraan menggunakan metode *You Only Look Once* (YOLO) versi 5.

### **BAB 4            IMPLEMENTASI DAN PENGUJIAN**

Bab ini berisikan penjelasan mengenai implementasi sistem dan hasil pengujian terhadap beberapa aspek dari sistem yang telah dibuat.

### **BAB 5            KESIMPULAN DAN SARAN**

Bab ini memuat kesimpulan dari penelitian yang telah dilakukan beserta saran dari peneliti yang bisa digunakan sebagai acuan penelitian selanjutnya.



## **BAB 2**

### **LANDASAN TEORI**

#### **2.1 Kendaraan**

Menurut Pasal 1 Angka 7 UU Nomor 22 Tahun 2009 Lalu Lintas dan Angkutan Jalan, kendaraan adalah suatu sarana angkut di jalan yang terdiri atas kendaraan bermotor dan kendaraan tidak bermotor. Kendaraan bermotor adalah setiap kendaraan yang digerakkan oleh peralatan mekanik berupa mesin selain kendaraan yang berjalan di atas rel, sedangkan kendaraan tidak bermotor setiap kendaraan yang digerakkan oleh tenaga manusia dan/atau hewan (Amwin, 2021). Selanjutnya, Pasal 1 Angka 7 UU Nomor 22 Tahun 2009 menjelaskan bahwa sepeda motor, mobil penumpang, mobil bus, mobil barang, dan kendaraan khusus termasuk ke dalam jenis kendaraan bermotor, sedangkan kendaraan yang digerakkan oleh tenaga manusia (seperti becak dan sepeda) dan kendaraan yang digerakkan oleh tenaga hewan (seperti delman) termasuk ke dalam jenis kendaraan tidak bermotor.

#### **2.2 Deteksi Objek**

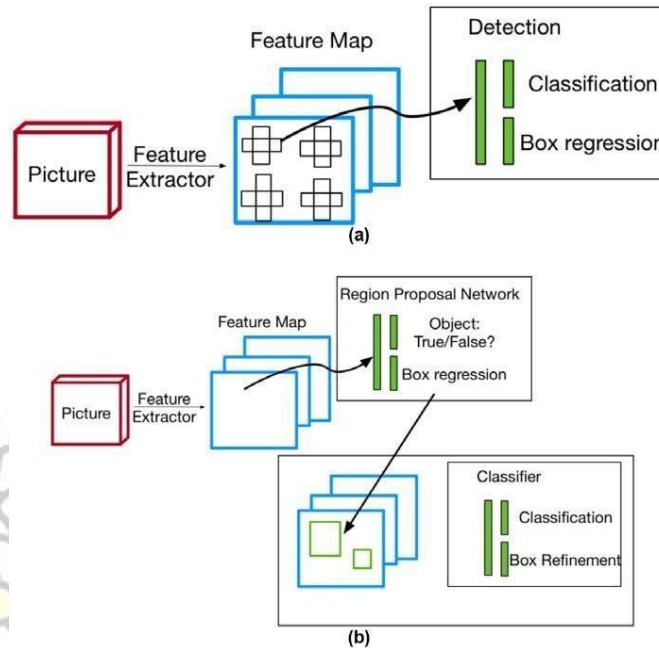
Salah satu metode utama dalam visi komputer, subbidang kecerdasan buatan yang mempelajari bagaimana mesin dapat memahami gambar seperti halnya manusia, adalah deteksi objek (Aningtiyas dkk., 2020). Menurut Zou dkk. (2019), deteksi objek merupakan "tugas visi komputer penting yang berhubungan dengan pendeteksian contoh objek visual dari kelas tertentu (seperti manusia, hewan, atau mobil) dalam gambar digital". Dengan kata lain, deteksi objek adalah proses mengidentifikasi atau menemukan contoh kelas objek visual tertentu (seperti manusia, hewan, dan kendaraan) dalam gambar digital. Tujuan deteksi objek, topik mendasar dalam visi komputer, adalah untuk membuat model dan metode komputasi yang dapat menjawab pertanyaan paling mendasar yang dibutuhkan oleh aplikasi visi komputer, khususnya: Objek apa yang ada di mana? (Zou dkk., 2019). Selain itu, deteksi objek mencakup sejumlah tugas visi komputer, termasuk pelacakan objek, pemberian teks pada gambar, dan segmentasi contoh.



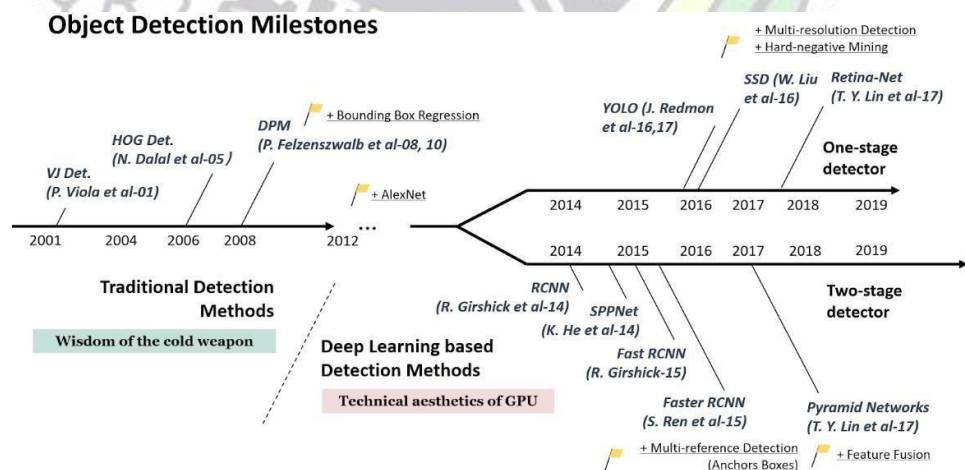
**Gambar 2.1.** Deteksi objek pada kendaraan *autonomous*  
(Sumber: Bin Issa, 2021)

Dalam dua dekade terakhir, deteksi objek telah melalui dua periode yaitu “periode deteksi objek tradisional (sebelum 2014)” dan “periode deteksi objek berbasis *Deep Learning* (sesudah 2014)” (Zou et al., 2019). Berdasarkan Zou et al. (2019), “periode deteksi objek tradisional” adalah periode dimana deteksi objek dilakukan dengan menggunakan metode-metode yang terkesan tradisional karena pada saat itu belum ditemukan representasi citra yang efektif sehingga mengharuskan seseorang untuk mendesain representasi fitur yang rumit dan berbagai kemampuan yang mampu mempercepat proses komputasi untuk mengurangi penggunaan sumber daya yang pada saat itu tentunya relatif terbatas, sedangkan “periode deteksi objek berbasis *Deep Learning*” adalah periode dimana deteksi objek dilakukan dengan menggunakan metode-metode yang lebih efektif dan efisien karena model deteksi objek berbasis *Deep Learning* dapat mempelajari representasi fitur yang rumit dan bertingkat tinggi. *Deep Learning* adalah salah satu bidang pada *Machine Learning* (ilmu yang memungkinkan komputer mampu mengembangkan perilaku berdasarkan data empiris, seperti data sensor atau basis data) yang menggunakan banyak lapisan pada pengolahan informasi nonlinier dalam melakukan ekstraksi fitur, pengenalan pola, dan klasifikasi (Amwin, 2021).

Pada era deteksi objek berbasis *Deep Learning*, deteksi objek terbagi menjadi dua kategori yaitu “*Two-stage Detection*” dan “*One-stage Detection*” (Zou et al., 2019).



**Gambar 2.2.** Plot skema (a) deteksi objek “*One-stage Detector*” dan (b) deteksi objek “*Two-stage Detector*” (Sumber: Kemajou et al., 2019)



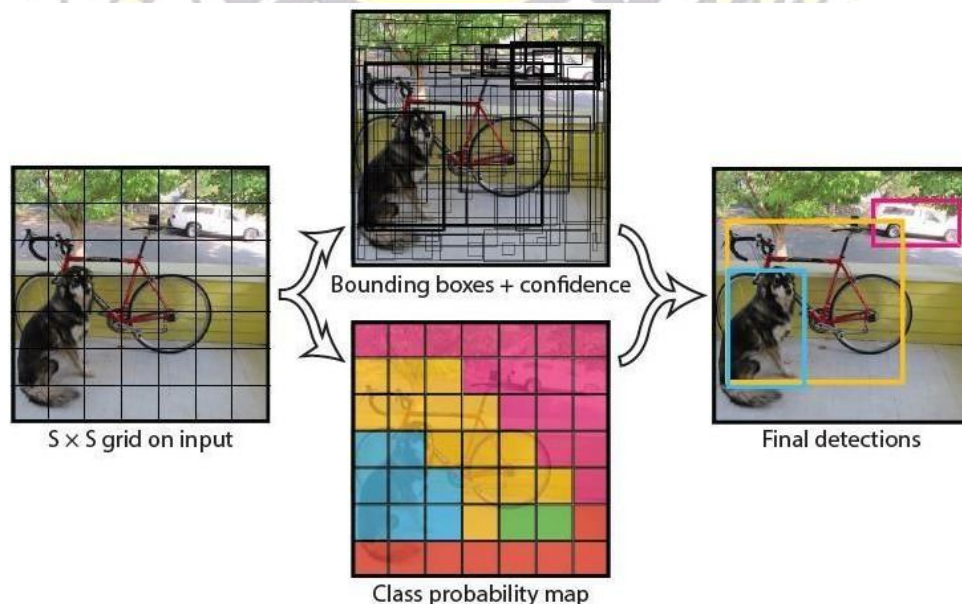
**Gambar 2.3.** Road map perkembangan deteksi objek selama dua dekade terakhir (Sumber: Zou et al., 2019)



### 2.3 Metode *You Only Look Once* (YOLO)

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

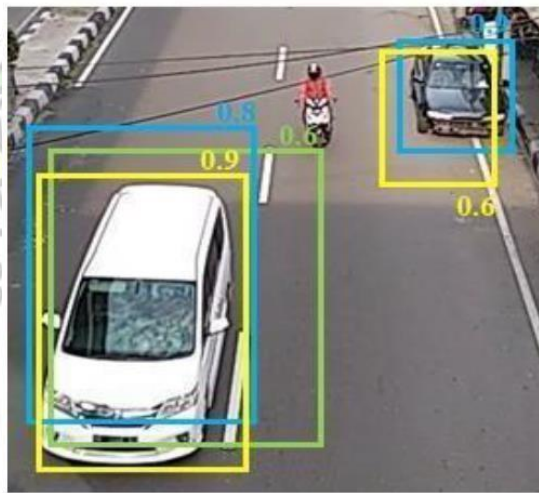
Sesuai dengan namanya, algoritma *You Only Look Once* (YOLO), berbeda dengan algoritma-algoritma deteksi objek sebelumnya yang menganut paradigma “deteksi proposal (kandidat *Bounding Boxes*) + verifikasi (klasifikasi dan lokalisasi objek)”, YOLO hanya menggunakan satu lapisan jaringan syaraf (*Neural Network*) pada citra. Jaringan ini akan membagi citra menjadi beberapa *regions* (daerah) dan memprediksi *bounding boxes* dan probabilitas setiap *regions* secara bersamaan (Zou et al., 2019). Walaupun memiliki kecepatan deteksi yang lebih cepat, YOLO membuat lebih banyak kesalahan lokalisasi objek, hal ini disebabkan karena YOLO tidak melakukan tahap deteksi proposal terlebih dahulu. Selain itu, YOLO juga kesusahan dalam mendeteksi objek berukuran kecil dan berdempetan.



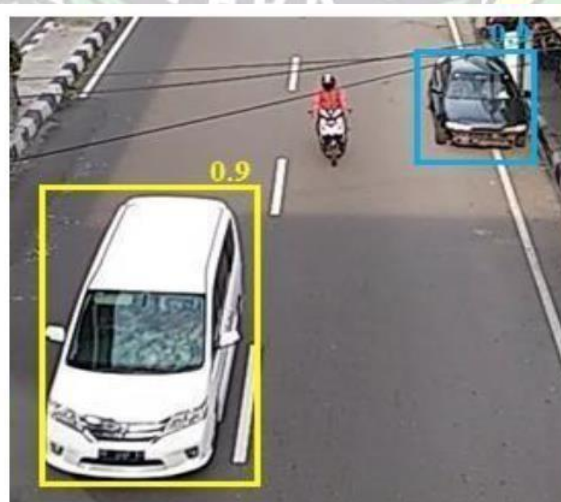
**Gambar 2.4.** Konsep pendeteksian algoritma YOLO

(Sumber: Redmon et al., 2016)

Dalam proses deteksi objek, algoritma YOLO akan mendeteksi beberapa objek besar atau objek yang berdekatan dengan batas sel-sel *grid* dalam banyak *bounding box* yang akurasi lokalisasinya beragam (seperti pada Gambar 2.5.), dari yang rendah (nilai IOU mendekati 0) dan tinggi (nilai IOU mendekati 1). Hal ini tentunya tidak diinginkan dan untuk mengatasi itu, dapat dilakukan operasi *Non-maximal Suppression*. *Non-maximal Suppression* (NMS) adalah operasi pembuangan *bounding box* yang memiliki nilai IOU kurang dari *threshold* (nilai ambang) yang ditentukan seperti pada Gambar 2.5. dan Gambar 2.6.



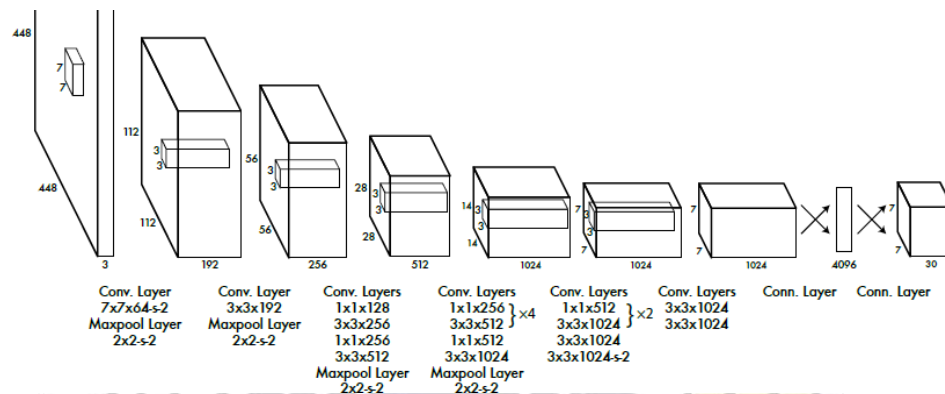
**Gambar 2.5.** Prediksi *bounding box* yang beragam (Leriansyah, 2020)



**Gambar 2.6.** Pembuangan *bounding box* yang nilainya kurang dari *threshold* yang ditentukan (Leriansyah, 2020)



Pada Gambar 2.5., dapat dilihat pada citra tersebut, terdapat beberapa *bounding box* pada objek mobil yang mempunyai nilai IOU yang beragam. Dengan NMS, *bounding box* yang berlebihan ini dapat dibuang. Pada Gambar 2.6., ditentukan *Threshold* sebesar 0,9, dengan begitu *Bounding Box* yang memiliki nilai IOU kurang dari 0,9 akan dibuang.



**Gambar 2.7.** Arsitektur YOLO (Redmond et al., 2016)

Setelah satu minggu pelatihan dengan *framework Darknet*, jaringan YOLO mencapai akurasi 88% pada ImageNet 2012 *Validation Set*. 20 lapisan *Convolutional* awal digunakan untuk proses *pretraining*. ImageNet 1000-*class Competition Dataset* digunakan untuk proses *pretraining* ini.

Menurut Jiang et al. (2022), YOLO telah berkembang dalam sejumlah cara, diantaranya adalah:

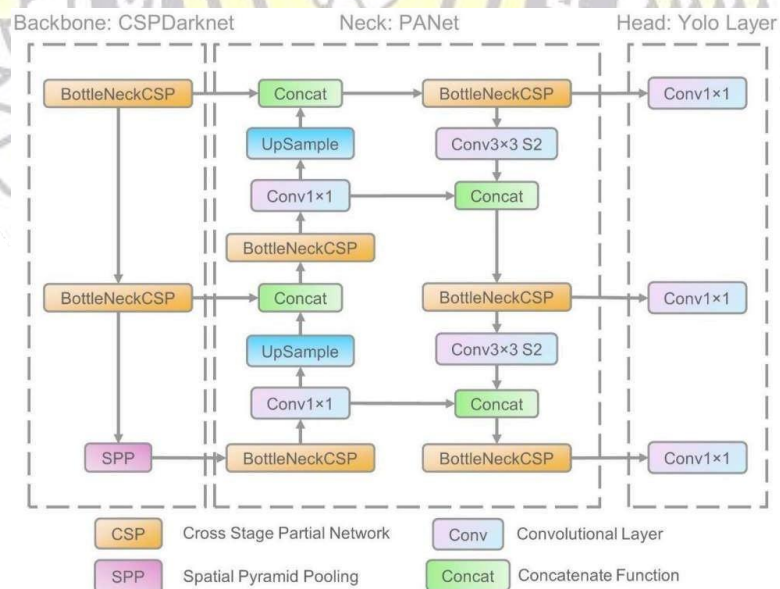
- YOLO v2 (2016) yang menambahkan *anchor* dengan K-means, *Two-stage Training*, serta *Full Convolutional Network*.
- YOLO v3 (2018) yang menambahkan deteksi *Multi-scale* dengan menggunakan FPN.
- YOLO v4 (2019) yang menambahkan SPP, fungsi aktivasi MISH, *Data Enhancement Mosaic/Mixup*, GIOU (*Generalized Intersection over Union*) *Loss Function*.
- YOLO v5 (2020) menambahkan kontrol lebih fleksibel terhadap ukuran model, pengaplikasian fungsi aktivasi Hardswish & *Data Enhancement*.

Selain beberapa versi YOLO di atas terdapat juga beberapa versi YOLO lain yang memiliki kelebihan serta kekurangannya masing-masing seperti YOLO 9000, Tiny YOLO, Fast YOLO, dan YOLACT (*You Only Look at Coefficients*).

Joseph Redmond, yang meninggalkan komunitas AI setelah pertama kali memulai algoritma YOLO, tidak terlibat aktif dalam pengembangan algoritma YOLO v4 atau iterasi selanjutnya dari algoritma YOLO.

## 2.4 *You only Look Once (YOLO) v5*

YOLO v5 diperkenalkan oleh Glenn Jocher menggunakan *framework Pytorch*. Sama seperti versi sebelumnya YOLO v5 didasarkan pada arsitektur deteksi YOLO seperti *bounding box anchor*, augmentasi data *mosaic*, jaringan *partial cross-stage*, dan sebagainya. Tiap – tiap algoritma memiliki fungsi yang berbeda pada lokasi yang berbeda pada arsitektur YOLO v5 (Li et al., 2021). Arsitektur YOLO v5 terdiri atas tiga bagian utama, yaitu *backbone*, *neck* dan *head*.



**Gambar 2.8.** Arsitektur YOLOv5 (Xu et al., 2021)

#### 2.4.1. Backbone

Jaringan parsial *cross-stage* dimasukkan ke dalam *Darknet*, menghasilkan CSPDarknet sebagai *backbone*nya. Dengan CSPNet masalah seperti perulangan informasi gradien dalam *backbone* skala besar dapat diatasi. Selain CSPNet *backbone* juga mencakup struktur Fokus, dan SPP untuk mengekstrak *feature map* ukuran yang berbeda dari gambar input dengan beberapa konvolusi dan pooling. CSPNet juga mengintegrasikan perubahan gradien ke dalam *feature map* sehingga mengurangi parameter dan FLOPS model. CSPNet melakukan fusi lintas lapisan lokal, memanfaatkan informasi fitur dari lapisan yang berbeda untuk mendapatkan *feature map* yang lebih kaya.

#### 2.4.2. Neck

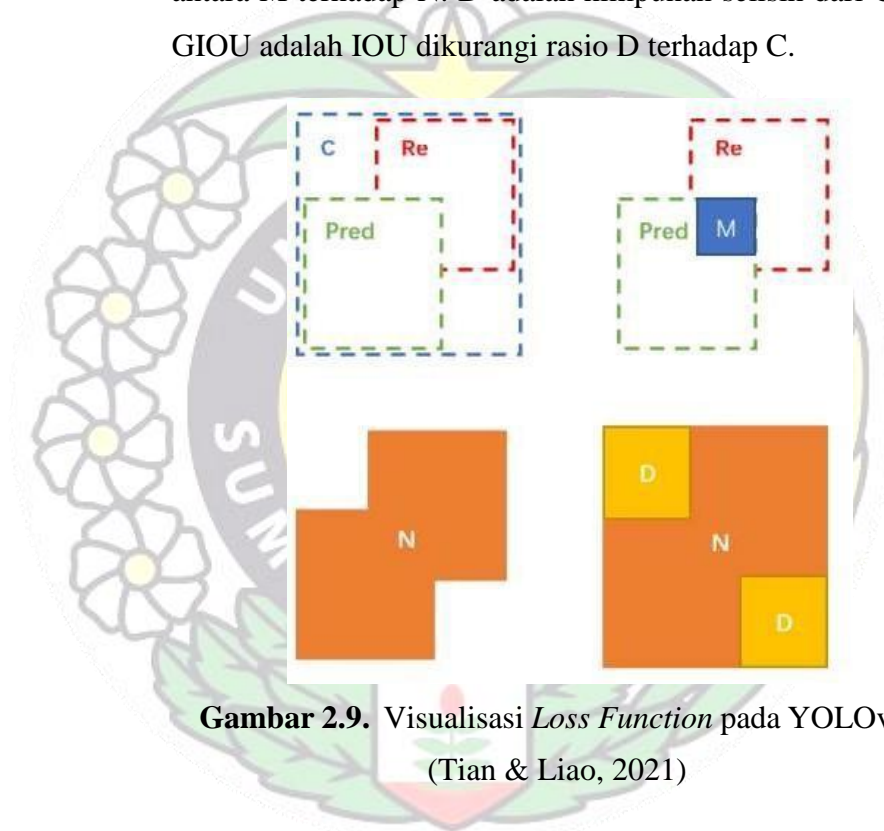
YOLOv5 menggunakan *Path Aggregation Network* (PANet) yang mengadopsi *feature pyramid network* baru dimana hal ini meningkatkan penyebaran fitur *low-level*. Neck terutama digunakan untuk menghasilkan piramida fitur, meningkatkan deteksi model objek dengan skala yang berbeda, dan menyadari pengenalan objek yang sama dengan ukuran dan skala yang berbeda. Ekstraktor fitur pada YOLOv5 menggunakan struktur FPN baru, yang meningkatkan jalur *bottom-up* dan meningkatkan penyebaran fitur tingkat rendah.

#### 2.4.3. Head

Terakhir, bagian *head* pada YOLOv5 yang disebut sebagai *yolo layer* menghasilkan kotak deteksi, menunjukkan kategori, koordinat, dan *confidence* dengan menerapkan *anchor box* ke *feature map* multiskala dari modul *neck* (Tian dan Liao, 2021) .

- *Loss Function*

*Loss function* YOLOv5 menggunakan GIOU\_Loss, yang meringankan kondisi ketika IOU\_Loss tidak dapat menangani dua box. Seperti yang terlihat pada Gambar 14, diasumsikan bahwa persegi panjang eksternal minimum dari *prediction box* dan *groundtruth box* adalah C, Himpunan dari *prediction box* dan *ground truth box* adalah N, Perpotongan *prediction box* dan *ground truth box* adalah M maka IOU adalah perbandingan antara M terhadap N. D adalah himpunan selisih dari C dan N. GIOU adalah IOU dikurangi rasio D terhadap C.



**Gambar 2.9.** Visualisasi *Loss Function* pada YOLOv5  
(Tian & Liao, 2021)

- *Non Maximum Suppression*

*Non Maximum Suppression* adalah tahap akhir dalam proses pendeteksian citra. Pada tahap ini algoritma akan menghapus *bounding box* yang redundan dan menyisakan hanya satu *bounding box* terbaik.



## 2.5 Penelitian Relevan

Adapun beberapa penelitian yang berkaitan dengan penelitian adalah:

1. Penelitian dilakukan oleh Wisna, J. S., Matulatan, T., & Hayaty, N. (2020) tentang Deteksi Kendaraan Secara Real Time Menggunakan Metode YOLO Berbasis Android menunjukkan bahwa pendeteksian kendaraan sepeda motor, mobil, truk, dan bus memiliki nilai akurasi pendeteksian sebesar 83.3%. Aplikasi deteksi kendaraan menampilkan *bounding box*, label kendaraan dan nilai keyakinan. Namun, data *training* terbatas dan kemampuan kamera berdampak signifikan pada nilai keyakinan yang dihasilkan oleh sistem deteksi kendaraan.
2. Penelitian dilakukan oleh Zhang, Y., Guo, Z., Wu, J., Tian, Y., Tang, H., & Guo, X.(2022) tentang *Real-Time Vehicle Detection Based on Improved YOLO v5* menghasilkan sistem pemantauan cerdas video lalu lintas dan deteksi kendaraan mencapai hasil yang baik di bidang jalan raya.
3. Penelitian yang dilakukan oleh Mulyana, D. I. & Rofik, M. A. (2022) tentang Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan metode YOLOv5 menyimpulkan hasil *training* pada penelitian pendeteksian jenis kendaraan di jalan raya yaitu nilai *precision* mendapatkan rata-rata nilai 0.995 terhadap nilai *recall*. Nilai puncak rata-rata *recall* mendapatkan nilai 1,00 pada nilai confidence 0,00. Dan juga hasil penelitian dapat disimpulkan nilai akurasi saat pengujian mendapatkan nilai yang baik yaitu sebesar 90% pada pendeteksian jenis kendaraan. Nilai akurasi dipengaruhi oleh kualitas video, kualitas dataset, dan pengambilan gambar diberbagai sudut.
4. Penelitian yang dilakukan oleh Dwiyanto, R., Widodo, D. W., & Kasih, P. (2022) tentang Implementasi Metode *You Only Look Once* (YOLOv5) Untuk Klasifikasi Kendaraan Pada CCTV Kabupaten Tulungagung menghasilkan penggunaan metode *You Only Look Once* (YOLO) dalam klasifikasi kendaraan memiliki performa memadai dalam mendeteksi kendaraan telah diuji yaitu motor, mobil, truk, dan bus. Tingkat akurasi yang tercatat adalah 79,8% pada siang hari, 74,7% pada sore hari, dan 41,5% pada malam hari.



## BAB 3

### ANALISIS DAN PERANCANGAN SISTEM

#### 3.1 Data yang digunakan

Data yang dipergunakan dalam penelitian ini merupakan data video yang diambil secara langsung menggunakan kamera *smartphone* dengan resolusi 50MP dan kemudian dikonversi menjadi .jpg. Proses pengumpulan data ini berlangsung kurang lebih 2 minggu dari Jalanan di Kota Medan. Data berformat .jpg yang memiliki ukuran piksel 640 x 640, totalnya berjumlah 300 data. Dataset yang digunakan terdiri dari 10 video yang diubah menjadi .jpg. Data yang sudah dikumpulkan dipisah ke dalam 3 jenis, yaitu data *training*, data *validation* dan sebagai data *testing*. Pembagian tersebut bertujuan untuk melakukan akurasi yang baik dalam melakukan deteksi kendaraan. Data *training* terdiri dari 260 data yang digunakan untuk melatih sistem, 26 data untuk validasi dalam pengelompokkan jenis kendaraan dan data untuk *testing* terdiri dari 14 data yang digunakan untuk menguji hasil data *testing* sebelum modelnya dibuat.

#### 3.2 Arsitektur Umum

Arsitektur umum merupakan skema penggambaran secara menyeluruh mengenai alur kerja penelitian hingga sistem selesai. Adapun arsitektur umum dari penelitian ini digambarkan sebagai berikut.

Tahap pertama penelitian adalah proses pengumpulan data. Data yang dipakai adalah video yang diambil dari Jalanan di Kota Medan menggunakan kamera dan kemudian dikonversi menjadi jpg. Jenis data citra yang digunakan ada lima, yaitu citra mobil, motor, bus, becak dan truck. Data yang dikumpulkan kemudian dibagi kedalam tiga kelompok yaitu data *training*, data *validation* dan data *testing*.

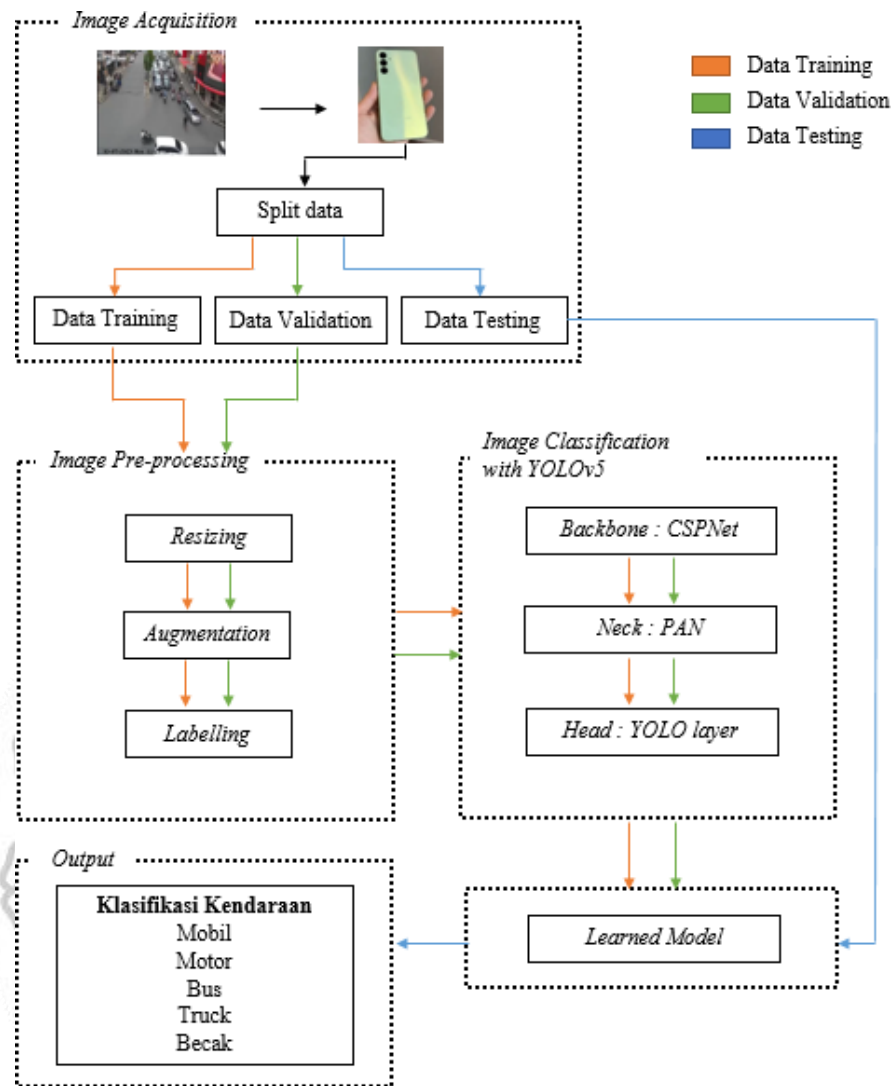
Tahap selanjutnya yaitu *pre-processing*, yaitu proses mempersiapkan data sebelum dilakukan *training*. Yang pertama yaitu *resizing*, proses menelaraskan ukuran citra yang akan diklasifikasi sehingga sama

seluruhnya. Setelah *resize* citra kemudian diaugmentasi untuk menambah kuantitas data. Kemudian citra yang telah diaugmentasikan diberi anotasi untuk memudahkan pengenalan citra.

Tahap terakhir yang dilakukan adalah klasifikasi dengan menerapkan algoritma *You Only Look Once* (YOLO) versi 5. YOLOv5 adalah *one-stage object detector* yang mampu mengidentifikasi objek dengan menggambarkan *bounding box* dan mengklasifikasikan objek berdasarkan probabilitas kelas yang ditetapkan. *One-stage object detector* arsitektur YOLOv5 terdiri atas tiga bagian, yaitu *backbone*, *neck*, dan *head*. Pada bagian *backbone* berisi CSPNet. Dengan CSPNet masalah seperti perulangan informasi gradien dalam *backbone* skala besar dapat diatasi. CSPNet mengintegrasikan perubahan gradien ke dalam *feature map* sehingga mengurangi parameter dan FLOPS model. CSPNet melakukan fusi lintas lapisan lokal, memanfaatkan informasi fitur dari lapisan yang berbeda untuk mendapatkan *feature map* yang lebih kaya. YOLOv5 menggunakan *Path Aggregation Network* (PANet) yang mengadopsi *feature pyramid network* baru dimana hal ini meningkatkan penyebaran fitur *low-level*. *Neck* digunakan untuk menghasilkan piramida fitur, meningkatkan deteksi model objek dengan skala berbeda, dan menyadari pengenalan objek yang sama dengan ukuran dan skala berbeda. Bagian *head* pada YOLOv5 disebut sebagai *yolo layer* menghasilkan kotak deteksi, menunjukkan kategori, koordinat & *confidence* dengan menerapkan *anchor box* ke *feature map* multiskala dari modul *neck* (Tian dan Liao, 2021).

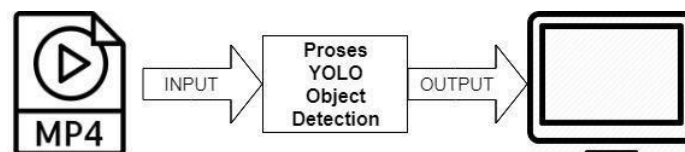
Setelah proses *training* dan klasifikasi selesai akan dihasilkan file akhir berupa .py yang merupakan *learned model*. *Learned model* adalah output hasil *training* data atau hasil pembelajaran yang dilakukan oleh mesin selama proses training dimana nantinya model ini akan digunakan oleh data *testing* untuk menguji keakuratan pengklasifikasian.

Setelah menyelesaikan langkah-langkah ini, maka menghasilkan lima *output* klasifikasi kendaraan yang ada di Kota Medan antara lain: mobil, motor, bus, becak dan truck. Pada gambar ditampilkan arsitektur umum yang menampilkan tahap-tahap analisis sistem.



**Gambar 3.1.** Arsitektur Umum Penelitian

### 3.3 Pemodelan Sistem

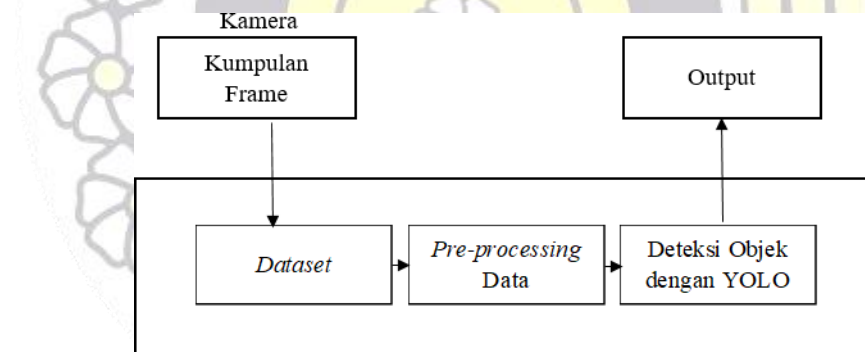


**Gambar 3.2.** Blok Diagram Aplikasi Penghitung Kendaraan

Gambar blok diagram diatas, memperoleh dari visual yang sudah diambil dari video yang direkam di jalanan kota Medan akan diproses oleh *image processing* pada *Roboflow* menggunakan metode *YOLO object detection* untuk menghitung berapa banyak kendaraan yang berada pada salah satu ruas jalan. Program *YOLO Detection Object* hanya akan menghitung kendaraan jika kendaraan melewati garis berwarna biru yang berfungsi menghitung kendaraan yang lewat.

### 3.4 Sistem Pre-Processing Data

Kumpulan frame yang menjadi sebuah gambar yang berurutan selanjutnya akan melalui beberapa tahapan, mulai dari pengolahan data gambar sampai deteksi objek dengan menggunakan algoritma YOLO. Agar lebih mudah memahami tahapan dalam sistem *processing* data ini, maka dibuatkan diagram blok yang ditunjukkan pada gambar 3.3.

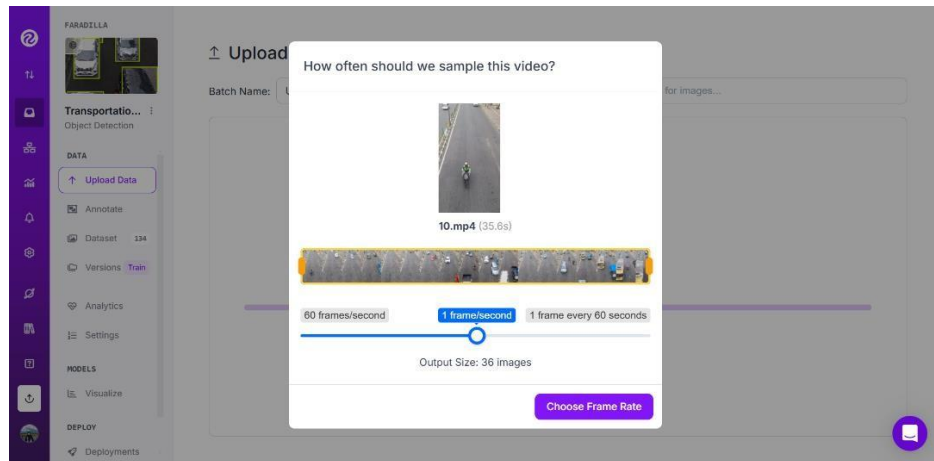


**Gambar 3.3.** Diagram Blok Sistem *Processing* Data

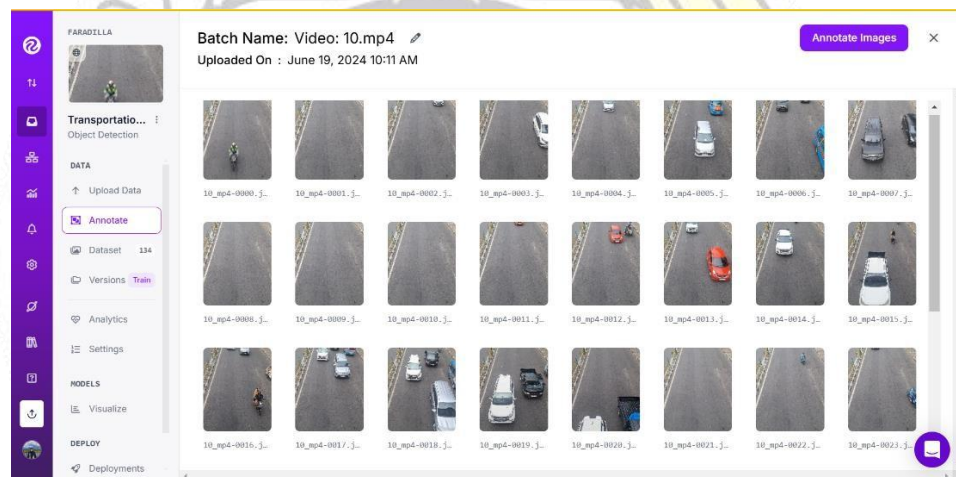
#### 3.4.1. Akuisisi Data

Akuisisi data atau *datasheet* merupakan suatu kumpulan data objek yang berupa gambar dari objek yang akan dideteksi, pada penelitian ini objek yang akan dideteksi yaitu kendaraan. Tahap akuisisi data ini, kumpulan data diperoleh dari suatu video yang dihasilkan oleh kamera *smartphone* yang di *extract* menjadi beberapa frame sehingga menjadi gambar yang berurutan. Proses *convert* video menjadi sebuah gambar ini menggunakan *Roboflow*.





**Gambar 3.4.** Tampilan *Roboflow* Fitur *Convert Video to JPG*

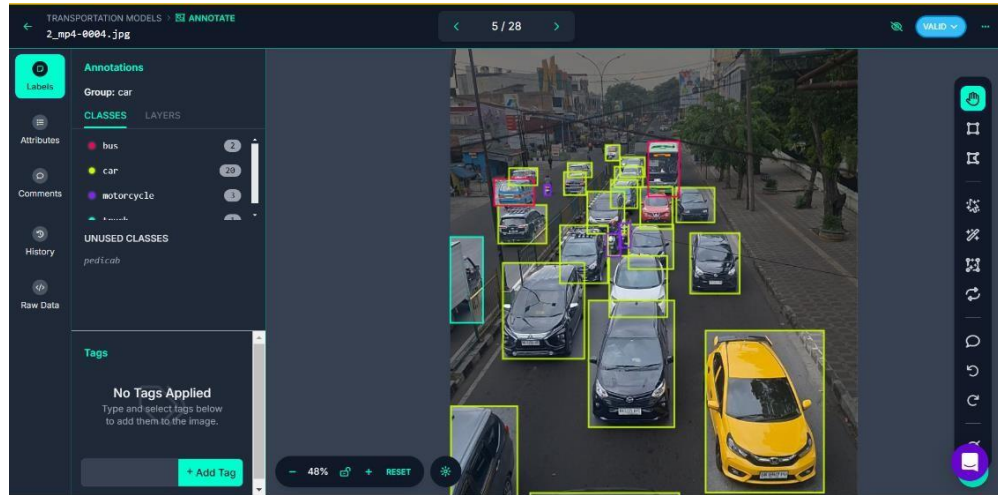


**Gambar 3.5.** Hasil *Convert Video to JPG* sebelum dilakukan pelabelan

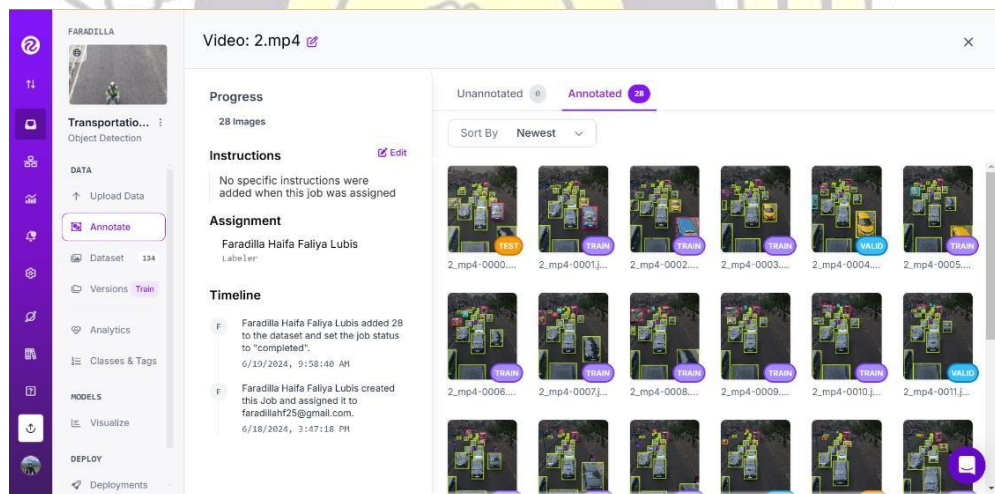
### 3.4.2. *Pre-Processing Data*

Video yang telah di ekstrak menjadi kumpulan gambar yang berurutan kemudian dilanjut pada tahap *pre-processing* data. Tahap ini terdiri dari duatahapan yaitu tahap *resizing* dan *cropping* serta anotasi data. Kedua tahap ini dilakukan untuk keperluan proses lebih lanjut yang akan diinputkan pada algoritma YOLO. Proses anotasi data yang dilakukan yaitu dengan memberikan keterangan pada objek berupa kotak pembatas/*box*, kelas dan skor kepercayaan. Proses anotasi ini akan menghasilkan data file dengan format *txt*. Berdasarkan tahapan *pre-processing* yang dilakukan menggunakan

laptop dapat dilanjutkan dengan 3 proses selanjutnya yaitu *train*, *test*, dan validasi. Pemberian label pada objek dengan menggunakan *Roboflow* dapat dilihat pada gambar 3.6.



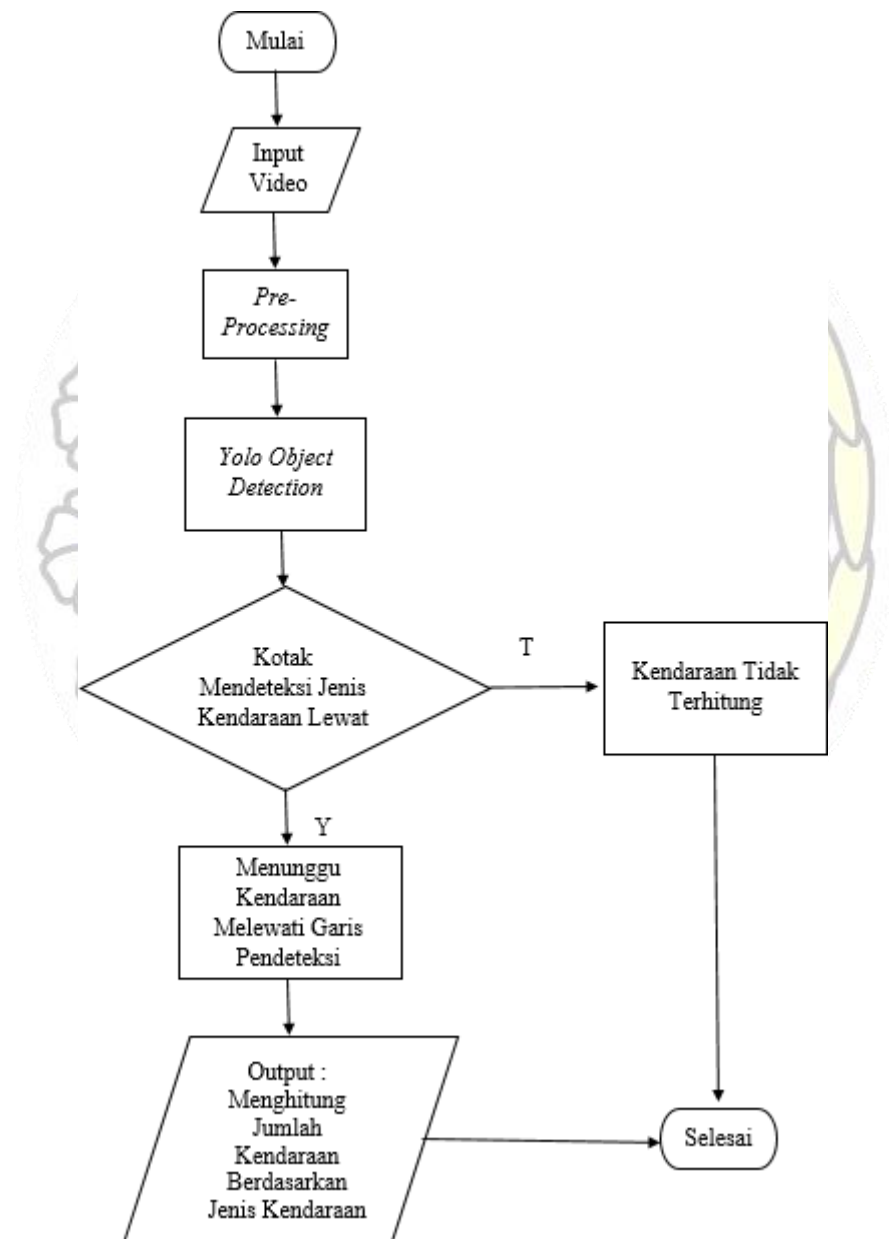
**Gambar 3.6.** Pemberian label pada objek



**Gambar 3.7.** Beberapa datasheet yang telah melalui proses pelabelan

### 3.5 Flowchart

Salah satu dari banyak langkah yang diperlukan untuk membuat program perangkat lunak lebih mudah dibuat adalah menggambar *flowchart* yang menunjukkan cara kerja program tersebut. Berikut ini *flowchart* yang ditampilkan.



**Gambar 3.8.** Flowchart Sistem

Berikut adalah penjelasan dari Gambar 3.8 *Flowchart* Sistem.

1. Input Rekaman Video ialah ketika memasukan rekaman video yang dibutuhkan program untuk mendeteksi kendaraan yang lewat.
2. *Pre-Processing* ialah proses mengklasifikasi objek gambar secara manual menggunakan *software*. User dapat mengklasifikasi semua objek dan menyimpan hasil gambar beserta *.txt* ke folder data sebelum masuk ke program.
3. YOLO Object Detection ialah ketika program sudah memulai bekerja mendeteksi kendaraan dan membuat garis untuk menghitung kendaraan yang melewatinya.
4. Kotak mendeteksi kendaraan ialah ketika program sudah berhasil mendeteksi kendaraan yang akan melintas.
5. Menunggu kendaraan melewati garis ialah ketika garis masih menunggu kendaraan yang akan melewatinya.
6. Menghitung jumlah kendaraan ialah ketika kendaraan yang sudah terdeteksi melewati garis dan counter otomatis akan bertambah.



## BAB 4

### IMPLEMENTASI DAN PENGUJIAN

#### 4.1 Kebutuhan Sistem

Untuk implementasi sistem, dibutuhkan kebutuhan perangkat yang terdiri dari perangkat lunak dan perangkat keras sebagai berikut.

##### 4.1.1 Perangkat Lunak

- a. *Framework* Visual Studio Code
- b. *Roboflow*
- c. Bahasa Pemrograman *Python*

##### 4.1.2 Perangkat Keras

- a. *Intel Core i5*
- b. RAM 8 GB
- c. *Harddisk* 1TB

#### 4.2 Hasil Akuisisi Data

Dataset yang dikumpulkan diambil dari rekaman video jalanan yang diambil di salah satu sudut jalan di Kota Medan. Rekaman video tersebut kemudian di ekstrak dengan menggunakan *software* Roboflow, sehingga menjadi kumpulan frame yang akan digunakan untuk proses *training* sistem algoritma YOLOv5 agar dapat mendeteksi suatu objek dari suatu kendaraan yang melintas dengan harapan didapat nilai *Confidence* yang cukup tinggi agar diperoleh nilai error seminimal mungkin. Dataset yang digunakan sebanyak 10 vidio, dimana dari 10 vidio diubah ke 300 gambar tersebut yang diambil di siang hari. Berdasarkan 300 gambar tersebut, dipecah kembali menjadi 3 bagian dengan perbandingan 88% (260 gambar untuk *Training* : 8% (26 gambar untuk Validasi) : 4% (14 gambar untuk *Testing*).

Setelah dataset terkumpul kemudian dilanjut dengan pemberian label (keterangan) pada objek yang akan dideteksi dengan pemberian bonderis (kotak pembatas) dan keterangan kelas objeknya. Kelas yang digunakan dalam pembuatan sistem ini hanya menggunakan lima kelas yaitu mobil,

motor, bus, truck dan becak. Proses pemberian label pada dataset ini menggunakan software labelImg. Proses anotasi ini akan menghasilkan file dengan format (.txt). Dalam file tersebut berisi nilai-nilai diantaranya kelas, x center, y center, lebar dan tinggi. File tersebut digunakan untuk melatih algoritma YOLOv5 agar dapat mengenali objek dari kedua kelas yang digunakan tersebut.

#### 4.3 *Training Dan Validasi*

Proses *training* dengan menggunakan algoritma YOLOv5 agar dapat mengenali objek yang akan dideteksi ini dilakukan dengan menggunakan sebuah situs berbasis web yaitu google colab atau google collaboratory. Google colab merupakan executable dokumen yang dapat digunakan untuk menulis, menyimpan serta membagikan suatu program yang telah ditulis melalui google drive. Google colab merupakan tempat yang cocok bagi programmer yang ingin mengasah pengetahuan mengenai bahasa pemrograman *python*. Selain gratis, google colab ini dapat dijalankan menggunakan google chrome, mozilla dan sebagainya. Google colab juga memiliki kumpulan *library machine learning*, serta terdapat GPU (*General Processing Unit*) yang dapat melakukan proses *training*, sehingga tidak memerlukan waktu yang cukup lama dalam memproses suatu gambar. Langkah awal yang dilakukan dalam proses *training* ini yaitu proses kloning data YOLOv5 pada situs github ultralytic YOLOv5. Adapun karakteristik dari YOLOv5 yaitu memiliki 213 layer/lapisan dengan 7.225.885 parameter, Batch yang dapat digunakan dapat diatur dari ukuran 16, 24, 40, serta epoch 100, 300 dan 500. Dalam mendeteksi objek berupa jenis kendaraan yaitu mobil, motor, truk, bus, dan becak algoritma YOLOv5 memiliki teknologi yang disebut dengan IOU(*Intersection Over Union*) dan *Non-Max Suppression*. Teknologi tersebut digunakan untuk mengukur rasio kotak pembatas pada objek yang akan di prediksi dengan anotasi kebenaran dasar saat  $IOU > 0,5 - 0,9$  dianggap dapat diterima. Dalam hal ini objek dengan nilai *Confidence* diatas 0,5 akan diberikan sebuah kotak pembatas pada objeknya, namun jika nilai *Confidence*

dibawah 0,5 maka akan dianggap sebagai *background* atau daerah tersebut dinyatakan tidak ada sebuah objek yang terdeteksi.

mAP (*mean average Precision*) merupakan suatu matrik yang umum digunakan dalam mengukur tingkat akurasi dalam mendeteksi suatu objek yang mana nilai dari matrik ini yaitu antara 0 hingga 1. Dapat dikatakan, semakin tinggi nilai atau skor dari mAP, maka sistem deteksi objek akan lebih akurat dalam pembacaannya. Dalam menghitung nilai mAP diperlukan IOU, *Precision*, Recall, *Precision Recall Curve*, dan AP.

Dalam dataset coco terdapat nilai mAP atau AP[0,5:0,05:0,95], dimana nilai 0,5 merupakan batas dari IOU. Nilai 0,05 merupakan penambahan nilai dari batas IOU hingga 0,95. Berdasarkan nilai yang diperoleh tersebut didapatkanlah nilai mAP. Nilai presisi berperan penting terhadap prediksi model dalam mendeteksi objek. Presisi dapat dihasilkan dengan persamaan sebagai berikut:

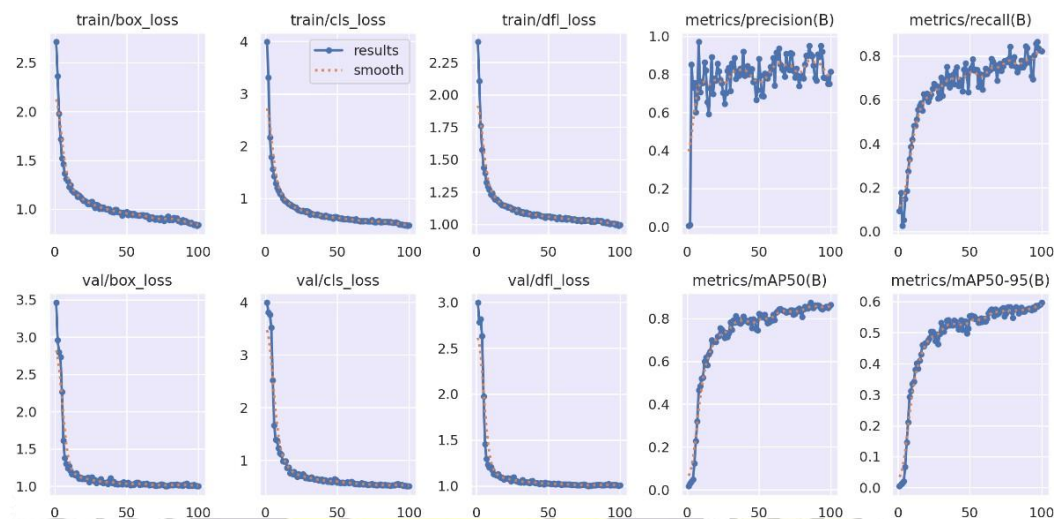
$$precision = \frac{TP}{TP+FP} \quad (4.1)$$

*True Positive* (TP) merupakan hasil yang diperoleh dari prediksi mesin yang menyatakan benar dan hal tersebut merupakan jawaban yang benar (*true*). *False Positive* (FP) merupakan hasil yang diperoleh berdasarkan prediksi mesin dinyatakan benar namun hal tersebut merupakan jawaban yang salah. Recall merupakan salah satu matrik yang digunakan untuk mengukur seberapa baik model yang telah dibuat dalam menemukan semua yang positif dalam kumpulan dataset yang diuji. Adapun persamaan dari matrik Recall dapat dituliskan sebagai berikut:

$$Recall = \frac{TP}{TP+FN} \quad (4.2)$$

*False Negatif* (FN) yang ada dalam persamaan Recall diatas merupakan hasil yang diperoleh dari prediksi oleh mesin yang dinyatakan salah namun merupakan jawaban yang benar (*true*). Matrik Recall ini menjadi acuan seberapa bagus model apabila kategori data tidak seimbang dan yang dapat berpengaruh besar terhadap lingkungan seperti halnya kesalahan prediksi penyakit yang menular.

Berdasarkan kinerja YOLOv5 yang telah dilatih dalam memprediksi suatu objek berupa kendaraan dengan pembagian kelas antara mobil, motor, truk, bus dan becak diperoleh nilai mAP keseluruhan di atas 80%. Berikut merupakan grafik dari kinerja YOLOv5 saat proses *training* ditunjukkan pada gambar 4.1.



**Gambar 4.1** Plot *Box*, *Objectness*, *Classification*, *Precision*, *Recall* setiap periode untuk *Training* dan *Validasi* Dataset

Gambar 4.1 menunjukkan grafik-grafik dari hasil *training* dan validasi, dimana pada grafik *Box* dan grafik *objectness* dapat dilihat menunjukkan penurunan yang cepat. Hal ini diakibatkan karena prediksi kotak pembatas tidak sepenuhnya menutupi objek-IOU yang salah. Grafik *classification* juga menunjukkan penurunan yang cepat, hal ini diakibatkan karena kerugian yang diakibatkan penyimpangan dari prediksi berdasarkan kelas yang telah ditentukan, misal objek berupa mobil diprediksi oleh sistem berupa motor. Grafik presisi dan grafik recall berkebalikan dengan grafik *box*, *objectness* dan *classification*. Grafik presisi dan recall mengalami kenaikan yang cepat. Hal ini karena presisi digunakan untuk mengukur seberapa akurat dari prediksi sistem, sedangkan recall merupakan suatu matrik yang digunakan untuk mengukur semua hal yang positif, jadi semakin tinggi grafik presisi dan recall semakin baik sistem tersebut bekerja.



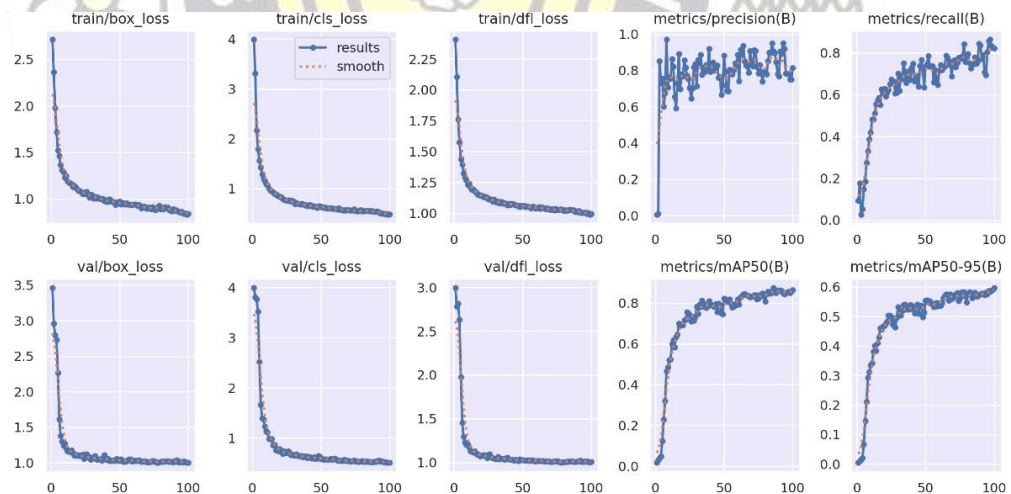
#### 4.4 Hasil Pengujian Algoritma YOLO Menggunakan Google Colab

Pengujian offline dihasilkan dari beberapa proses *training* yang dilakukan dengan menggunakan Google Colab. Tujuan dari pengujian offline ini yaitu untuk mengetahui kinerja dari algoritma YOLOv5 dalam memprediksi suatu objek.

**Tabel 4.1** Hasil pengujian google colab

NO	IMG	EPOCH	BATCH	MODEL	OPTIMIZER	mAP
1	640	100	32	YOLOV5	AUTO	0,831

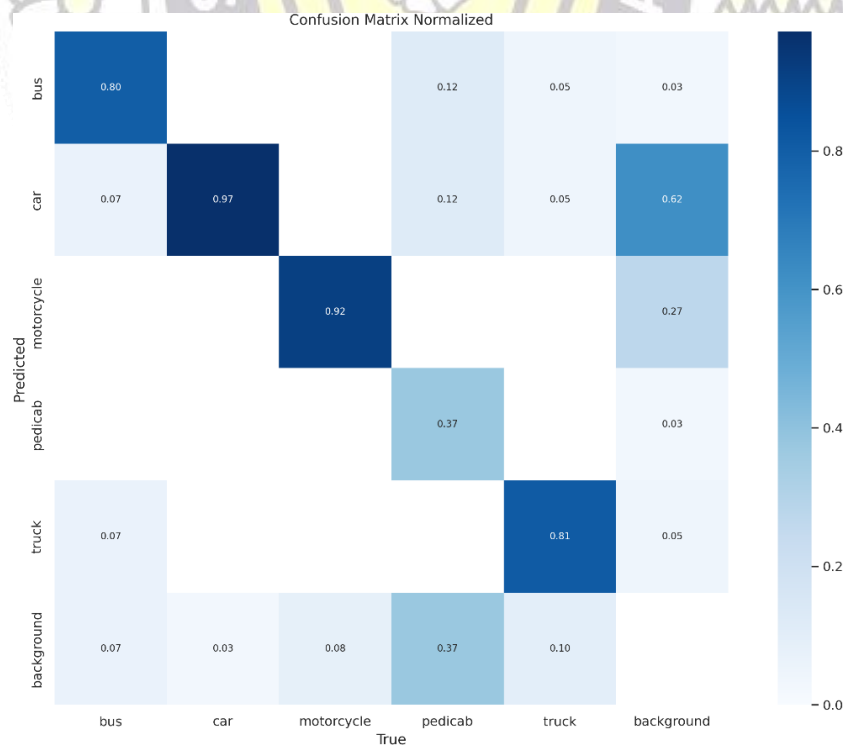
Pengujian model YOLOv5 juga dilakukan dalam proses *training*. Pengujian model YOLO dikombinasikan dengan beberapa hyperparameter seperti Epoch, Image, Batch, dan Optimizer. Berdasarkan pengujian dari model YOLOv5 diperoleh hasil *mean average Precision* (mAP) sebesar 0.831. Berikut merupakan grafik yang dihasilkan dari pengujian model YOLOV5 menggunakan parameter image size 640, Epoch 100, Batch 32, serta Optimizer AUTO dilihat pada gambar 4.2.



**Gambar 4.2** Hasil *training* menggunakan model YOLOv5

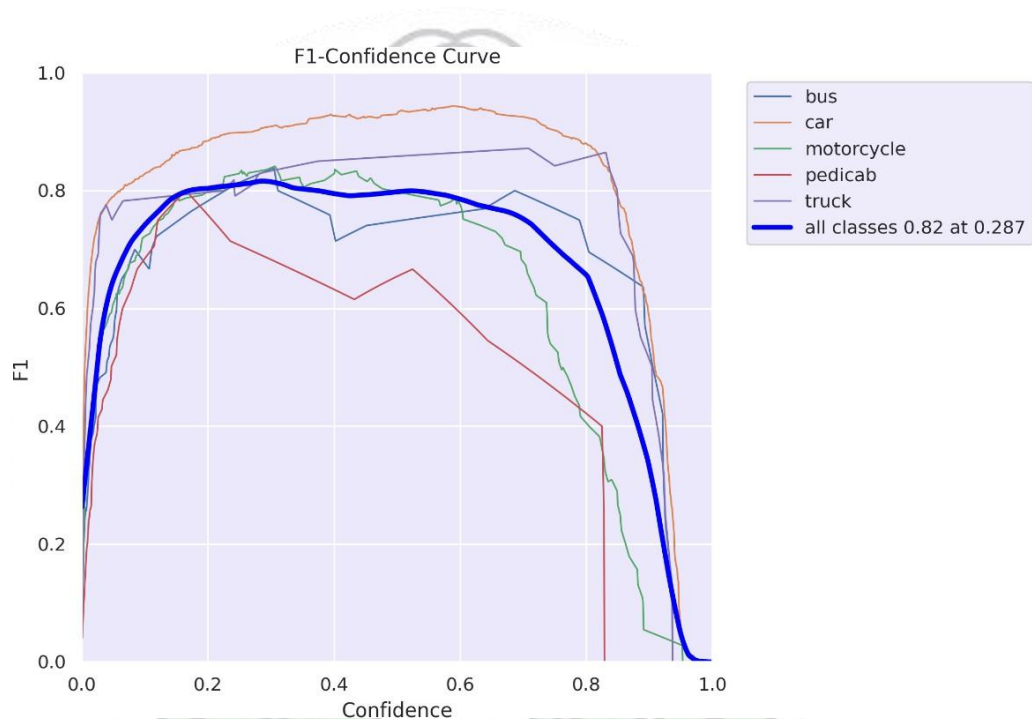
Berdasarkan gambar 4.2 diatas dihasilkan beberapa grafik *training* dan validasi diantaranya grafik bounding box, objek, kelas, mAP, *Precision* dan Recall. Berdasarkan grafik box\_loss, objek\_loss dan class\_loss mengalami penurunan secara eksponensial, hal ini dipengaruhi oleh banyaknya Epoch pada saat melakukan *training*. Semakin banyaknya Epoch yang dijalankan

maka box dan objek akan dihasilkan loss yang semakin sedikit, namun terlalu banyak Epoch juga tidak terlalu baik dari sistem yang akan dibuat, karena penurunan loss menunjukkan grafik eksponensial negatif sehingga perubahan loss yang dialami ketika Epoch semakin besar tidak terlalu signifikan dan hanya akan memberatkan sistem dan menghabiskan banyak waktu ketika proses *training*. Sehingga dipilih Epoch dengan nilai 100, karena dianggap nilai tersebut merupakan nilai yang paling optimal dan tidak terlalu memberatkan sistem serta nilai loss yang dihasilkan tidak terlalu besar. Selanjutnya untuk grafik mAP 50, mAP 50:95, *Precision* dan *Recall* kebalikan dari ketiga grafik sebelumnya. Ketika semakin banyak Epoch yang terlatih, maka untuk nilai dari grafik mAP, *Recall* dan *Precision* akan semakin meningkat atau mendekati nilai 1.0. Semakin besar nilai yang diperoleh dari ketigagrafik tersebut, maka dapat dikatakan sistem yang dibuat semakin baik. Adapun nilai yang diperoleh dari grafik mAP, *Precision* dan *Recall* yaitu untuk mAP 50 sebesar 0.83, mAP 50:95 sebesar 0.57, nilai *Precision* diperoleh sebesar 0.87 dan *Recall* sebesar 0.70.



**Gambar 4.3** Confusion Matrix

Gambar 4.3 merupakan hasil dari Confusion matrix yang dihasilkan dari proses training dan validasi pada saat pengujian model YOLOv5. Confusion matrix di atas menunjukkan bahwa YOLOv5 dapat memprediksi suatu kelas diantaranya mobil dengan akurasi rata-rata sebesar 97%, memprediksi kelas motor dengan akurasi rata-rata sebesar 92%, memprediksi kelas bus dengan akurasi rata-rata sebesar 80%, memprediksi kelas truk dengan akurasi rata-rata sebesar 81% dan memprediksi kelas becak dengan akurasi rata-rata sebesar 37%.

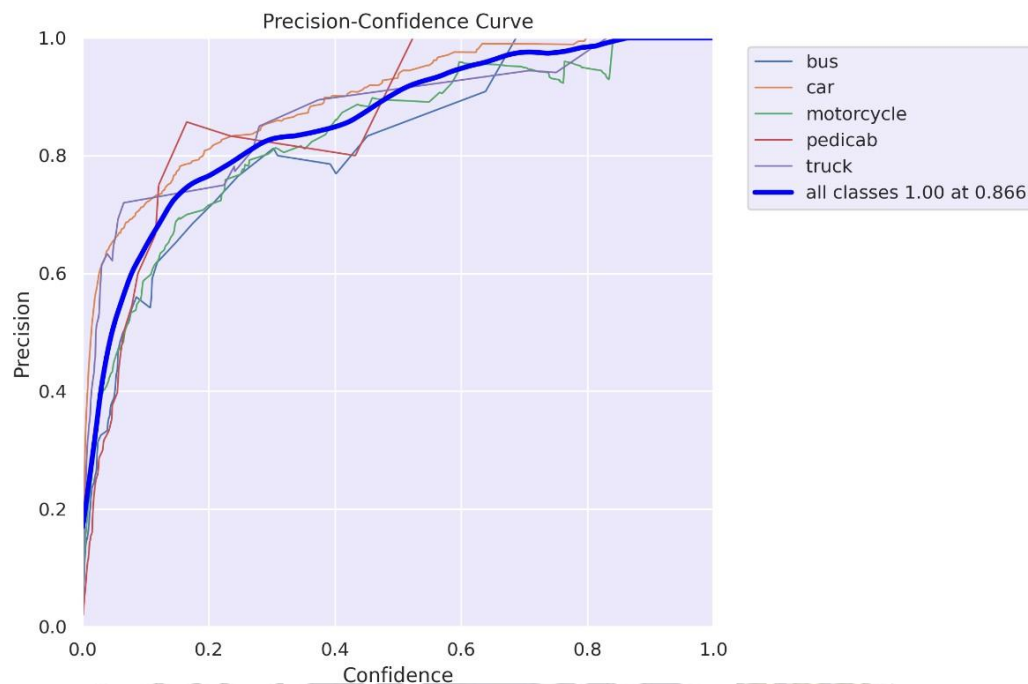


**Gambar 4.4** Kurva nilai F1 terhadap nilai *Confidence*

Gambar 4.4 diatas menunjukkan grafik dari F1 Score. F1 Score merupakan ukuran akurasi model pada kumpulan data yang diberikan dan merupakan data yang diperoleh dari kombinasi nilai *Precision* dan *Recall*. Adapun rumus dari F1 Score sendiri dapat dilihat seperti pada persamaan dibawah ini:

$$F1\ Score = \frac{2x(Precision \times Recall)}{Precision + Recall} \quad (4.3)$$

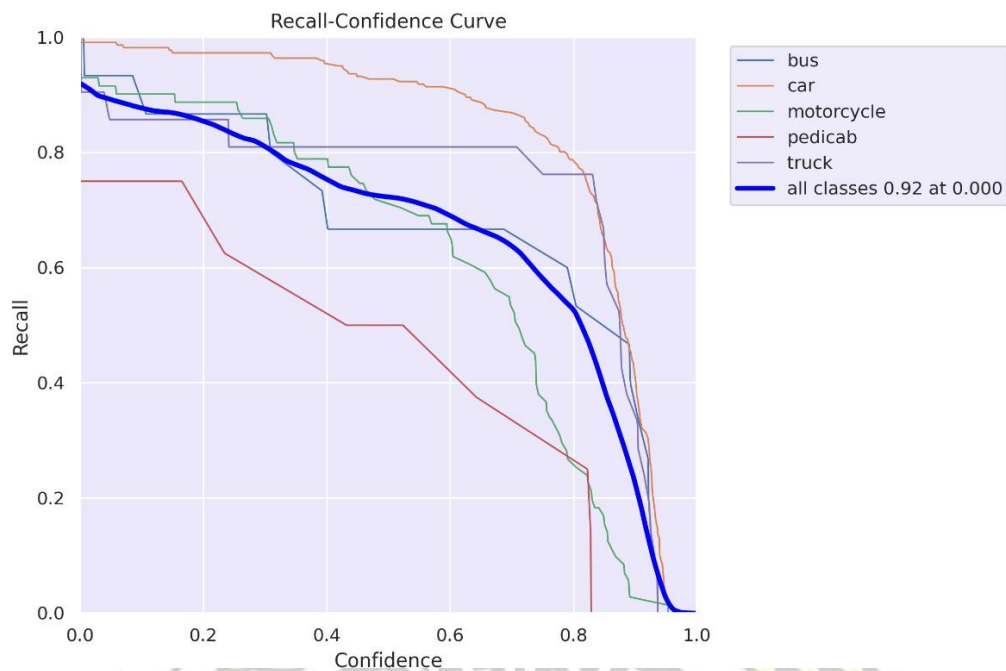
Nilai F1 Score mencapai titik tertinggi sebesar 0.82 setiap kelasnya pada saat nilai *Confidence* 0.287.



**Gambar 4.5** Kurva nilai *Precision* terhadap nilai *Confidence*

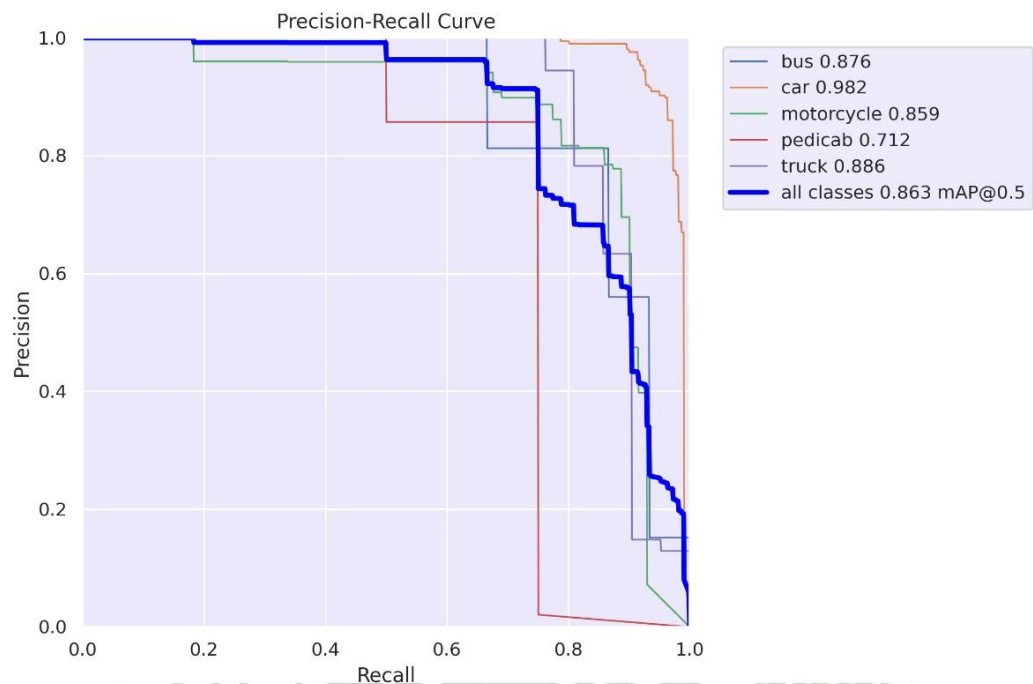
Gambar 4.5 menunjukkan grafik antara nilai *Confidence* terhadap *Precision*. Dapat dilihat pada grafik yang dihasilkan dimana kelas kendaraan mencapai nilai *Precision* tertinggi pada saat *Confidence* skornya berada pada nilai 0.866. Berdasarkan dari grafik yang dihasilkan dapat dikatakan bahwa semakin cepat nilai kendaraan mencapai puncak tertingginya maka, sistem yang dibuat akan semakin baik dalam memprediksi suatu objek. Nilai *Confidence* tersebut menunjukkan seberapa baik sistem tersebut dalam memprediksi suatu objek.





**Gambar 4.6** Kurva nilai Recall terhadap nilai *Confidence*

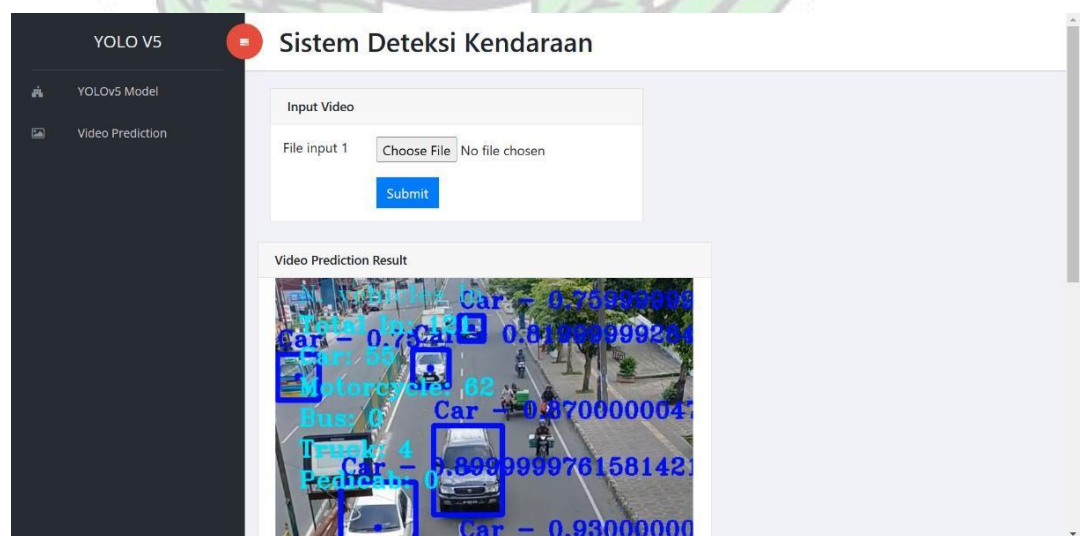
Gambar 4.6 menunjukkan grafik antara *Confidence* dengan nilai Recall yang diperoleh setelah melakukan proses *training* dan validasi sistem. Grafik tersebut menunjukkan nilai antara masing-masing kelas mobil dan motor mencapai nilai Recall 0.92 pada saat *Confidence* skornya 0. Saat *Confidence* Skor nilainya mendekati 1.0, maka nilai Recall yang dari masing-masing kelas akan menurun. Hal ini karena nilai Recall merupakan suatu matrik yang digunakan untuk mengukur suatu hal yang positif.



**Gambar 4.7** Kurva nilai *Precision* terhadap nilai Recall

Gambar 4.7 menunjukkan grafik antara nilai *Precision* dan Nilai Recall. Berdasarkan dari grafik yang dihasilkan dapat dilihat bahwa nilai tertinggi dari kelas bus 0.876, mobil 0.982, motor 0.859, becak 0.712, dan truck 0.886 pada saat nilai mAP 0.5.

#### 4.5 Hasil Pengujian Penghitung Kendaraan Yang Melintas



**Gambar 4. 8** Hasil Hitung Kendaraan Melintas

Pada pengujian kali ini program YOLO sudah dapat menghitung kendaraan. Ditandai dengan adanya bantuan garis biru yang berfungsi untuk menghitung banyaknya kendaraan yang melewatinya dan hasilnya akan ditampilkan di layer pada bagian pojok kiri.

Selanjutnya adalah dengan pengambilan data dengan cara mengamati berapa banyak mobil yang lewat setiap 10 detik sampai durasi video habis dan melihat kecocokan antara counter aplikasi dengan hitungan manual. Proses pengambilan data dilakukan menggunakan 4 sample video yang berbeda dari berbagai sumber yang ada pada internet serta rekaman penulis. Adapun data dari hasil analisa adalah sebagai berikut :

**Tabel 4.2** Hasil pengujian pada video 1 (Durasi 40 Detik)

Nomor	Detik	Jumlah Kendaraan												Total Deteksi Error
		Terdeteksi						Manual						
		Mobil	Motor	Bus	Truck	Becak	Total	Mobil	Motor	Bus	Truck	Becak	Total	
1	ke-10	1	1	-	-	-	2	2	4	-	-	1	7	3
2	ke-20	6	2	-	-	-	8	6	6	-	-	1	13	
3	ke-30	9	9	-	-	-	18	7	12	-	-	1	20	
4	ke-40	11	12	-	-	-	23	8	17	-	-	1	26	

Pada Tabel 4.2, hasil pengambilan data menunjukkan bahwa terdapat sejumlah kesalahan deteksi saat membandingkan akurasi penghitungan aplikasi dengan penghitungan manual. Meskipun sejumlah data yang berhasil telah diperoleh, kita dapat melanjutkan untuk menghitung akurasi.

$$\begin{aligned}
 \text{Keakuratan Data} &= 100\% - (\text{deteksi error} / \text{perhitungan manual} \times 100\%) \\
 &= 100\% - \left( \frac{3}{26} \times 100\% \right) \\
 &= 100\% - 12\% = 88\%
 \end{aligned}$$

**Tabel 4.3** Hasil pengujian pada video 2 (Durasi 40 Detik)

Nomor	Detik	Jumlah Kendaraan												Total Deteksi Error
		Terdeteksi						Manual						
		Mobil	Motor	Bus	Truck	Becak	Total	Mobil	Motor	Bus	Truck	Becak	Total	
1	ke-10	-	11	1	-	-	12	-	6	1	1	1	9	2
2	ke-20	-	11	1	-	-	12	-	6	1	1	1	9	
3	ke-30	-	11	1	-	-	12	-	7	1	1	1	10	
4	ke-40	-	11	1	-	-	12	-	7	1	1	1	10	

Pada Tabel 4.3, hasil pengambilan data menunjukkan bahwa terdapat sejumlah kesalahan deteksi saat membandingkan akurasi penghitungan aplikasi dengan penghitungan manual. Meskipun sejumlah data yang berhasil telah diperoleh, kita dapat melanjutkan untuk menghitung akurasi.

$$\begin{aligned}\text{Keakuratan Data} &= 100\% - (\text{deteksi error/perhitungan manual} \times 100\%) \\ &= 100\% - \left(\frac{2}{10} \times 100\%\right) \\ &= 100\% - 20\% = 80\%\end{aligned}$$

**Tabel 4.4** Hasil pengujian pada video 3 (Durasi 40 Detik)

Nomor	Detik	Jumlah Kendaraan												Total Deteksi Error
		Terdeteksi						Manual						
		Mobil	Motor	Bus	Truck	Becak	Total	Mobil	Motor	Bus	Truck	Becak	Total	
1	ke-10	4	1	-	-	1	6	4	3	-	1	1	9	17
2	ke-20	7	1	-	-	1	9	7	3	-	1	1	12	
3	ke-30	7	6	-	-	1	14	7	11	-	1	1	20	
4	ke-40	10	12	-	-	1	23	10	28	-	1	1	40	

Pada Tabel 4.4, hasil pengambilan data menunjukkan bahwa terdapat sejumlah kesalahan deteksi saat membandingkan akurasi penghitungan aplikasi dengan penghitungan manual. Meskipun sejumlah data yang berhasil telah diperoleh, kita dapat melanjutkan untuk menghitung akurasi.

$$\begin{aligned}\text{Keakuratan Data} &= 100\% - (\text{deteksi error/perhitungan manual} \times 100\%) \\ &= 100\% - \left(\frac{17}{40} \times 100\%\right) \\ &= 100\% - 42\% = 57\%\end{aligned}$$

**Tabel 4.5** Hasil pengujian pada video 4 (Durasi 1 Menit 50 Detik)

Nomor	Detik	Jumlah Kendaraan												Total Deteksi Error
		Terdeteksi						Manual						
		Mobil	Motor	Bus	Truck	Becak	Total	Mobil	Motor	Bus	Truck	Becak	Total	
1	ke-10	2	8	-	-	-	10	2	10	-	-	1	13	7
2	ke-20	7	9	-	-	-	16	6	13	-	-	1	20	
3	ke-30	14	12	-	-	-	26	10	18	-	-	1	29	
4	ke-40	17	18	-	-	-	35	12	27	-	-	1	40	
5	ke-50	18	24	-	-	-	42	13	35	-	-	1	49	
6	ke-60	20	27	-	-	-	47	15	40	-	-	2	57	
7	ke-70	24	27	-	-	-	51	16	41	-	-	3	60	
8	ke-80	25	33	-	-	1	59	17	44	-	-	4	65	
9	ke-90	35	49	-	-	1	85	20	52	-	-	5	77	
10	ke-100	43	54	-	3	1	101	26	59	-	2	5	92	
11	ke-110	55	62	-	4	1	122	40	65	-	3	5	113	



Pada Tabel 4.5, hasil pengambilan data menunjukkan bahwa terdapat sejumlah kesalahan deteksi saat membandingkan akurasi penghitungan aplikasi dengan penghitungan manual. Meskipun sejumlah data yang berhasil telah diperoleh, kita dapat melanjutkan untuk menghitung akurasi.

$$\begin{aligned}
 \text{Keakuratan Data} &= 100\% - (\text{deteksi error/perhitungan manual} \times 100\%) \\
 &= 100\% - \left( \frac{7}{113} \times 100\% \right) \\
 &= 100\% - 6\% = 94\%
 \end{aligned}$$

Tabel 4.2 sampai Tabel 4.5 merupakan hasil uji sistem menggunakan 4 video uji dari total data uji yang digunakan terdapat 189 kendaraan dengan 1 bus, 58 mobil, 117 motor, 8 becak dan 5 truck . Dari 4 video data uji tersebut, terlihat sistem melakukan beberapa kali kesalahan dalam mendeteksi. Untuk mengukur performa keseluruhan sistem dapat dilakukan dengan menggunakan *confusion matrix*. *Confusion matrix* akan memberikan hasil prediksi dari data yang diuji pada sistem dengan menampilkan nilai dari *true positive*, *false positive*, *true negative*, dan *false negative*. *Confusion matrix* hasil prediksi jenis kendaraan pada penelitian ini dapat dilihat pada Tabel 4.6.

**Tabel 4.6.** *Confusion matrix* hasil uji sistem

	<i>bus</i>	<i>car</i>	<i>motorcycle</i>	<i>pedicab</i>	<i>truck</i>
<i>Bus</i>	<b>1</b>	0	0	0	0
<i>Car</i>	0	<b>76</b>	14	0	0
<i>motorcycle</i>	0	17	<b>97</b>	0	0
<i>pedicab</i>	0	0	6	<b>2</b>	0
<i>truck</i>	0	1	0	0	<b>4</b>



Pada Tabel 4.6 dapat dilihat bahwa sistem mampu mendeteksi 1 bus, 76 mobil, 97 motor, 2 becak, dan 4 truck dengan benar. Dapat dilihat juga bahwa sistem melakukan beberapa kesalahan dalam mendeteksi. 14 mobil terdeteksi motor oleh sistem, 17 motor terdeteksi mobil oleh sistem, 6 becak terdeteksi motor oleh sistem dan 1 truck terdeteksi mobil oleh sistem. Berdasarkan *confusion matrix* pada tabel terlihat bahwa sistem yang dibangun pada penelitian ini mampu melakukan deteksi bus secara sempurna.

Dari Tabel 4.6 juga kita dapat menghitung nilai *true positive*, *false positive*, *true negative*, dan *false negative* dari masing – masing kelas atau jenis kendaraan yang digunakan pada penelitian. Pengertian dari nilai – nilai tersebut adalah :

1. *True positive* adalah hasil yang menunjukkan nilai aktual positif dan sistem berhasil mendeteksi citra dengan nilai positif. Artinya sistem berhasil mendeteksi jenis kendaraan dengan benar.
2. *False positive* adalah hasil yang menunjukkan nilai aktual negatif namun sistem mendeteksi dengan nilai positif.
3. *True negative* adalah nilai yang menunjukkan nilai aktual negatif dan sistem benar mendeteksi dengan nilai negatif.
4. *False negative* adalah nilai yang menunjukkan nilai aktual positif namun sistem mendeteksi dengan nilai negatif.

Nilai *true positive*, *false positive*, dan *false negative* untuk masing – masing jenis kendaraan yang digunakan pada penelitian ini berdasarkan hasil uji sistem dapat dilihat pada Tabel 4.7.

**Tabel 4.7.** Nilai *true positive*, *false positive*, *true negative*, dan *false negative* hasil uji

	TP	FP	TN	FN
<b>Bus</b>	1	0	188	0
<b>Car</b>	76	18	81	14

<i>motorcycle</i>	97	20	55	17
<i>Pedicab</i>	2	0	181	6
<i>Truck</i>	4	0	184	1

Berdasarkan nilai *false positive* Tabel 4.7 dapat dilihat sistem memprediksi 18 citra bukan mobil sebagai mobil dan 20 citra bukan motor sebagai motor. Dari nilai *false negative* pada tabel, sistem memprediksi 14 citra mobil sebagai bukan mobil, 17 citra motor sebagai bukan motor, 6 citra becak sebagai bukan becak, dan 1 citra truck sebagai bukan truck.

Dari nilai – nilai *true positive*, *false positive*, dan *false negative* pada Tabel 4.7 kita dapat mengukur performa sistem seperti akurasi sistem, *precision*, *recall*, dan *F1 score* untuk masing – masing jenis kendaraan. Perhitungan dan penjelasan dari performa sistem yang dihitung pada penelitian ini, yaitu :

1. Akurasi merupakan persentase keakuratan sistem dalam mendeteksi objek, dalam hal ini objek tersebut merupakan citra kendaraan. Nilai akurasi dapat dihitung dengan membagi total data yang berhasil dideteksi sistem benar dengan jumlah total data uji. Perhitungan akurasi sistem dapat dilihat pada persamaan (4.4).

$$\begin{aligned}
 \text{Akurasi} &= \frac{\text{total data uji yang dideteksi dengan benar}}{\text{total data uji}} \times 100\% & (4.4) \\
 &= \frac{\text{total TP}}{\text{total data uji}} \times 100\% \\
 &= \frac{180}{189} \times 100\%
 \end{aligned}$$

$$\text{Akurasi} = 95 \%$$

2. *Precision* merupakan rasio nilai kelas yang diklasifikasi positif oleh sistem dengan total keseluruhan nilai yang diklasifikasi positif oleh sistem. Secara umum, perhitungan *precision* dapat dilakukan dengan persamaan (4.5).

$$Precision = \frac{TP}{TP+FP} \quad (4.5)$$

Menggunakan persamaan (4.5), nilai *precision* untuk bus, yaitu :

$$Precision = \frac{1}{1+0}$$

$$Precision = \frac{1}{1}$$

$$Precision = 1.00$$

Nilai *precision* untuk mobil, yaitu :

$$Precision = \frac{76}{76+18}$$

$$Precision = \frac{76}{94}$$

$$Precision = 0.80$$

Nilai *precision* untuk motor, yaitu :

$$Precision = \frac{97}{97+20}$$

$$Precision = \frac{97}{117}$$

$$Precision = 0.82$$

Nilai *precision* untuk becak, yaitu :

$$Precision = \frac{2}{2+0}$$

$$Precision = \frac{2}{2}$$

$$Precision = 1.00$$

Nilai *precision* untuk truck, yaitu :

$$Precision = \frac{3}{3+0}$$

$$Precision = \frac{3}{3}$$

$$Precision = 1.00$$

3. *Recall* merupakan rasio nilai kelas yang diklasifikasi positif oleh sistem dengan total keseluruhan data positif. Secara umum, *recall* dapat dihitung dengan persamaan (4.6).

$$Recall = \frac{TP}{TP+FN} \quad (4.6)$$

Menggunakan persamaan (4.6), nilai *recall* untuk bus, yaitu :

$$Recall = \frac{1}{1 + 0}$$

$$Recall = \frac{1}{1}$$

$$Recall = 1.00$$

Nilai *recall* untuk mobil, yaitu :

$$Recall = \frac{76}{76 + 14}$$

$$Recall = \frac{76}{90}$$

$$Recall = 0.84$$

Nilai *recall* untuk motor, yaitu :

$$Recall = \frac{97}{97 + 17}$$

$$Recall = \frac{97}{114}$$

$$Recall = 0.85$$

Nilai *recall* untuk becak, yaitu :

$$Recall = \frac{2}{2 + 6}$$

$$Recall = \frac{2}{8}$$

$$Recall = 0.25$$

Nilai *recall* untuk truck, yaitu :

$$Recall = \frac{3}{3 + 1}$$

$$Recall = \frac{3}{4}$$

$$Recall = 0.75$$

4. *F1 score* merupakan rata – rata bobot *precision* dan *recall*. *F1 score* merupakan metrik evaluasi yang mengkombinasikan nilai *precision* dan *recall* untuk memberikan gambaran hasil yang lebih besar. Secara umum, perhitungan *F1 score* dapat dilakukan dengan persamaan (4.7).

$$F1\ Score = \frac{2x(Precision \times Recall)}{Precision+Recall} \quad (4.7)$$

Berdasarkan persamaan (4.7), dapat dihitung *F1 score* untuk bus, yaitu:

$$F1\ Score = \frac{2 \times 1.00 \times 1.00}{1.00 + 1.00}$$

$$F1\ Score = \frac{2}{2}$$

$$F1\ Score = 1.00$$

*F1 score* untuk mobil, yaitu:

$$F1\ Score = \frac{2 \times 0.80 \times 0.84}{0.80 + 0.84}$$

$$F1\ Score = \frac{1.34}{1.64}$$

$$F1\ Score = 0.81$$

*F1 score* untuk motor, yaitu:

$$F1\ Score = \frac{2 \times 0.82 \times 0.85}{0.82 + 0.85}$$

$$F1\ Score = \frac{1.39}{1.67}$$

$$F1\ Score = 0.83$$

*F1 score* untuk becak, yaitu:

$$F1\ Score = \frac{2 \times 1.00 \times 0.25}{1.00 + 0.25}$$



$$F1\ Score = \frac{0.50}{1.25}$$

$$F1\ Score = 0.40$$

*F1 score* untuk truck, yaitu:

$$F1\ Score = \frac{2 \times 1.00 \times 0.75}{1.00 + 0.75}$$

$$F1\ Score = \frac{1.50}{1.75}$$

$$F1\ Score = 0.85$$

Tabel nilai *precision*, *recall*, dan *F1 score* untuk masing – masing jenis kendaraan dapat dilihat pada Tabel 4.8.



**Tabel 4.8.** *Precision*, *recall*, dan *F1 score* masing - masing kendaraan

	<i>Precision</i>	<i>Recall</i>	<i>F1 score</i>
<b><i>Bus</i></b>	1.00	1.00	1.00
<b><i>Mobil</i></b>	0.80	0.84	0.81
<b><i>Motor</i></b>	0.82	0.85	0.83
<b><i>Becak</i></b>	1.00	0.25	0.40
<b><i>Truck</i></b>	1.00	0.75	0.85

Berdasarkan hasil uji sistem yang sudah dijabarkan sebelumnya, dapat diketahui bahwa sistem mampu mendeteksi 180 kendaraan dengan benar dari total 189 data uji. Dari perhitungan akurasi dan perhitungan performa lainnya, didapati bahwa sistem yang mengimplementasikan metode *You Only Look Once* (YOLO) v5 ini memiliki akurasi sebesar 95%. Beberapa faktor yang mempengaruhi hal ini dikarenakan objek yang diteliti tidak melewati garis pendeteksi dan kondisi jalanan yang terlalu ramai.

#### 4.6 Pembahasan

Penelitian ini menggunakan metode YOLO (*You Only Look Once*) sebagai algoritma deteksi objek untuk menguji kemampuannya dalam mengenali berbagai jenis kendaraan yang melintas di lalu lintas Kota Medan. Dengan menggunakan metode YOLO (*You Only Look Once*) ini untuk melakukan deteksi kendaraan pada penelitian ini mampu memberikan hasil sebesar 95% dari sistem yang dibangun untuk kendaraan mobil, motor, bus, truk dan becak. Terdapat beberapa kesalahan yang dilakukan oleh sistem ketika melakukan deteksi pada video yang terekam di jalanan kota medan. Beberapa contoh dari kesalahan tersebut dapat dilihat pada Tabel 4.9.

Data Video	Label sebenarnya	Hasil deteksi
	<i>pedicab</i>	<i>motor</i>
	<i>motor</i>	<i>tidak terdeteksi</i>

Kesalahan yang dilakukan oleh sistem ini dipengaruhi oleh ciri fisik dari becak dan motor yang sangat mirip karena becak di medan merupakan motor yang dimodifikasi untuk mengangkut penumpang lebih banyak. Penelitian ini juga memiliki beberapa faktor lain yang mempengaruhi kesalahan tidak terdeteksinya kendaraan seperti kendaraan yang diteliti berada di situasi lalu lintas yang terlalu padat, kendaraan tidak melewati garis pendeteksi sehingga tidak terhitung dan intensitas cahaya dari video yang diambil.

## **BAB 5**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Beberapa kesimpulan dapat diambil dari hasil kajian, perancangan, implementasi, dan pengujian sistem pendeteksi dan penghitungan kendaraan bermotor dengan metode *You Only Look Once* (YOLO):

1. Sistem aplikasi penghitung kendaraan yang melintas di jalan raya menggunakan bantuan dari metode YOLO dimodifikasi sehingga tidak hanya mendeteksi kendaraan tapi juga dapat menghitung kendaraan yang melewatinya.
2. Hasil pengujian aplikasi menunjukkan bahwa metode YOLO mampu membedakan berbagai jenis kendaraan, termasuk motor, mobil, bus, truk, dan becak, dengan memberikan penanda berupa kotak berwarna biru beserta nama kendaraan tersebut di dalam frame video.
3. Dari hasil pengujian aplikasi penghitung kendaraan berdasarkan metode YOLO menggunakan 4 video uji dengan durasi video diatas 40 detik terdapat 180 kendaraan telah berhasil terdeteksi dengan benar dari total 189 kendaraan. Dari perhitungan akurasi dan perhitungan performa lainnya, didapati bahwa sistem yang mengimplementasikan metode *You Only Look Once* (YOLO) v5 ini memiliki akurasi sebesar 95%.

#### **5.2 Saran**

Dalam perancangan dan pengujian yang telah dijalankan oleh penulis, terdapat beberapa hal yang dapat ditambahkan supaya hasil perancangan lebih baik dari penulis, diantaranya adalah:

1. Membandingkan nilai akurasi dari aplikasi penghitung kendaraan berdasarkan metode *You Only Look Once* dengan metode-metode lainnya.
2. Dapat digunakan disaat intensitas cahaya rendah dengan bantuan rekaman kamera yang lebih baik.
3. Sistem yang dibuat terdapat kesalahan mendeteksi objek saat kondisi lalu lintas terlalu ramai. Diharapkan untuk penelitian berikutnya sistem dapat mendeteksi semua kendaraan walaupun dalam keadaan ramai.

## DAFTAR PUSTAKA

- Dio Riza Pratama, M., Priyatna, B., Shofiah Hilabi, S., & Lia Hananto, A. (2022). Deteksi Objek Kecelakaan Pada Kendaraan Roda Empat Menggunakan Algoritma YOLOv5. *Teknologi: Jurnal Sistem Informasi*, 12(2), 15–26. <https://doi.org/https://doi.org/10.26594/teknologi.v12i2.3260>
- Djurianto, W., Maulana, E., & Brilianto, C. (2021). Aplikasi Pendeteksi Objek Kendaraan Menggunakan *Intel Movidius Neural Compute Stick* dengan Algoritma *MobileNet-YOLO*. *Jurnal EECCIS*, 15(3), 92-97.
- Dwiyanto, R., Widodo, D. W., & Kasih, P. (2022). Implementasi Metode You Only Look Once (YOLOv5) Untuk Klasifikasi Kendaraan Pada CCTV Kabupaten Tulungagung. *Seminar Nasional Inovasi Teknologi*, 102-104.
- Efendi, M. Y. & Abidin, M. H. F. (2024). Implementasi Klasifikasi Jenis Kendaraan Indonesia Menggunakan YOLO. *Seminar Nasional Teknik Elektro, Sistem Informasi, dan Teknik Informatika FTETI- Institut Teknologi Adhi Tama Surabaya*, 441-448.
- 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(3), 47-53.
- Hutauruk, J. S. W., Matulatan, T., & Hayaty, N. (2020). Deteksi Kendaraan secara Real Time menggunakan Metode YOLO Berbasis Android. *Jurnal Sustainable: Jurnal Hasil Penelitian Dan Industri Terapan*, 9(1), 8-14. <https://doi.org/10.31629/sustainable.v9i1.1401>
- Ilmawati, R. & Hustinawati. (2022). YOLO v5 untuk Deteksi Nomor Kendaraan di DKI Jakarta. *Jurnal INFOKAM*, 10(1), 32-43.
- Kim, J., Kim, N., Park, Y. W., & Won, C. S. (2022). Object Detection and Classification Based on YOLO-V5 with Improved Maritime Dataset. *J. Mar. Sci. Eng.* 2022, 10, 377. <https://doi.org/10.3390/jmse10030377>



- Lutfiyani, A. (2021). Deteksi Dan Pengenalan Objek Dengan Model Machine Learning: Model YOLO. *CESS (Journal of Computer Engineering System and Science)*, 6(2), 192-199.
- Mulyana, D. I. & Rofik, M. A. (2022). Implementasi Deteksi Real Time Klasifikasi Jenis Kendaraan Di Indonesia Menggunakan Metode YOLOV5. *Jurnal Pendidikan Tambusai*, 6(3), 13971-13982.
- Nugraha, A., Walidani, A., Arochman, D., Fahrezi, M. N., Agat, S. A. H., & Rosyani, P. (2023). Systematic Literatur Review Mendeteksi Kendaraan Menggunakan Metode YOLO (You Only Look Once). *JRIIN: Jurnal Riset Informatika dan Inovasi*, 1(3), 559-562.
- Ramadhan, A. K., & Laksono, S. B. (2022). Rancang Bangun Aplikasi Deteksi Objek Untuk Menghitung Jumlah Pengunjung Restoran Berbasis Computer Vision. *Jurusan Informatika Fakultas Teknik Universitas Persada Indonesia Y.A.I Jakarta*, 7(1), 46-57.
- Riansyah, A., & Mirza, A. H. (2023). Pendeteksi Mobil Berdasarkan Merek dan Tipe Menggunakan Algoritma YOLO. *SMATIKA JURNAL*, 13(01), 43–52. <https://doi.org/10.32664/smatika.v13i01.719>
- Syahputra, M.A., Pinem, J., Lubis, A.A., & Denia, Y. (2024). Implementasi Algoritma YOLO Dalam Pengklasifikasian Objek Transportasi pada Lalu Lintas Kota Medan. *Jurnal Penelitian Mahasiswa*, 3(1), 13-23
- Utama, R. A., & ETP, Lussiana. (2021). Implementasi Metode YOLO Object Detector untuk Klasifikasi Jenis Kendaraan yang Melintas di Ruas Jalan. *Jurnal Ilmiah Komputasi*, 20(4), 601-608.
- Wisna, J. S., Matulatan, T., & Hayaty, N. (2020). Deteksi Kendaraan Secara Real Time Menggunakan Metode YOLO Berbasis Android. *Jurnal Sustainable : Jurnal Hasil Penelitian dan Industri Terapan*, 9(1), 8-14.
- Zhang, Y., Guo, Z., Wu, J., Tian, Y., Tang, H., & Guo, X. (2022). Real-Time Vehicle Detection Based on Improved YOLO v5. *.Sustainability* 2022, 14, 12274. <https://doi.org/10.3390/su141912274>.

## LAMPIRAN 1. CURRICULUM VITAE



### Faradilla Haifa Faliya Lubis

+6281396139046 | faradillahf25@gmail.com

#### Tentang Saya

---

Lulusan S1 Ilmu Komputer Universitas Sumatera Utara (USU). Memiliki IPK 3,72 dan menguasai bahasa pemrograman C++ serta PHP. Berpengalaman dalam membangun website mulai dari nol menggunakan framework Laravel.

Punya kepribadian yang gigih dan pantang menyerah. Memiliki manajemen waktu yang baik. Memiliki minat karir di bidang pemrograman, manajemen data serta bidang yang terkait lainnya.

#### Pendidikan

---

2016 - 2019 | SMAN 1 Tebing Tinggi | Jurusan Ilmu Pengetahuan Alam

- Aktif mengikuti kegiatan internal sekolah
- Jurusan yang relevan : Matematika, Kimia, Fisika dan Biologi

2019 - Sekarang | Universitas Sumatera Utara | Program Studi Ilmu Komputer

Sarjana Ilmu Komputer, 3.72 / 4.00

- Aktif mengikuti kegiatan internal program studi Ilmu Komputer seperti bergabung dalam organisasi kampus dan kepanitiaan acara kampus

#### Pengalaman Studi Independen

---

2022 | Studi Independen | NF Computer

Mampu menerapkan desain web, mengimplementasikan desain UI/UX, menggunakan Git dan Github dalam proses pengembangan software atau pembuatan dokumen, mampu menggunakan bahasa pemrograman Javascript dan PHP, Mampu mengimplementasikan rancangan entitas dan keterkaitan antar entitas menggunakan SQL, dan menerapkan akses basis data SQL, membangun aplikasi web dengan framework Laravel dan API.

#### Pengalaman Organisasi

---

2021 - 2023 | Anggota Dana Dan Usaha | IMILKOM USU

Bertanggung jawab mencari dana untuk kebutuhan organisasi

2021 - 2022 | Anggota Keputrian | UKMI AL-KHUWARIZMI USU

Bertanggung jawab membuat kajian wanita muslimah

#### Kemampuan Personal & Perangkat Lunak

---

**Kemampuan Personal :** Kerjasama Tim, Komunikatif, Adaptif, Teliti, Kemampuan Analisis, Manajemen Waktu

**Kemampuan Perangkat Lunak :** Microsoft Word, Microsoft Excel, Microsoft Powerpoint