IMPLEMENTASI ALGORITMA YOLO PADA MODUL KAMERA UNTUK DETEKSI JENIS DAN KECEPATAN KENDARAAN

SKRIPSI

Oleh

Moh. Faizin

NIM 181910201105

PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS JEMBER

2023

IMPLEMENTASI ALGORITMA YOLO PADA MODUL KAMERA UNTUK DETEKSI JENIS DAN KECEPATAN KENDARAAN

SKRIPSI

Diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk menyelesaikan Program Studi Teknik Elektro (S1) dan mencapai gelar Sarjana Teknik

Oleh

Moh. Faizin

NIM 181910201105

PROGRAM STUDI STRATA 1 TEKNIK ELEKTRO

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS JEMBER

2023

PERSEMBAHAN

Puji syukur kehadirat Allah SWT karena atas ni'mat Allah kita tetap hidup dengan iman dan islam yang melekat pada diri kita. Sholawat dan salam atas junjungan kita Nabi Muhammad SAW sebagai suri tauladan dan rahmat bagi seluruh alam semesta. Dengan segala kerendahan hati, penulis mempersembahkan skripsi ini kepada:

1. Kedua orang tua saya, Bapak Moh. Yasin dan Ibu Ruqiah yang telah membesarkan, mendoakan sekaligus mendidik saya menjadi sosok seperti saat ini serta adik saya Rhobiatul Adawia;
2. Dosen pembimbing, Bapak Ir. Khairul Anam, S.T., M.T., Ph.D., IPM dan Bapak Ir. Wahyu Muldayani, S.T., M.T. yang telah memberikan arahan dan bimbingan selama pengerjaan skripsi ini;
3. Dosen penguji, Bapak Ir. Widya Cahyadi, S.T., M.T. dan Bapak Ir. Arizal Mujibtamala Nanda Imron, S.T., M.T. yang telah memberikan kritik dan saran kepada penulis;
4. Bapak Dedy Wahyu Herdiyanto, S.T., M.T. selaku Dosen Pembimbing Akademik yang telah memberikan bimbingan dan arahan selama masa perkuliahan;
5. Seluruh Dosen Teknik Elektro Universitas Jember yang telah menyalurkan banyak ilmunya kepada penulis selama masa perkuliahan.

MOTTO

"Ihya' Ulumiddin itu bagaikan embun, dia akan menyejukkan setiap orang yang menikmati maknanya"

(Moh. Faizin)

"Apa yang kita perlukan untuk masa depan? Tangan yang akan bekerja lebih keras dari biasanya, Mata yang akan melihat dari biasanya, Otak yang akan selalu berinspirasi dari biasanya, serta Mulut yang selalu berdo'a"

(Privasi)

"Saat jatuh, satu-satunya cara untuk maju adalah bangkit!"

PERNYATAAN

Saya yang bertanda tangan dibawah ini:

Nama : Moh. Faizin

NIM : 181910201105

Menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "IMPLEMENTASI ALGORITMA YOLO PADA MODUL KAMERA UNTUK DETEKSI JENIS DAN KECEPATAN KENDARAAN" adalah benar-benar hasil karya sendiri, kecuali kutipan yang telah saya sebutkan sumbernya, belum pernah diajukan pada instansi mana pun dan bukan jiplakan. Saya bertanggung jawab penuh atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya unsur tekanan dan paksaan pihak manapun serta bersedia mendapat sanksi akademik jika ternyata dikemudian hari pernyataan ini tidak benar.

Jember, 24 Januari 2023

Yang menyatakan,

Moh. Faizin

NIM. 181910201105

SKRIPSI

IMPLEMENTASI ALGORITMA YOLO PADA MODUL KAMERA UNTUK DETEKSI JENIS DAN KECEPATAN KENDARAAN

Oleh:

Moh. Faizin

181910201105

Pembimbing:

Dosen Pembimbing Utama : Ir. Wahyu Muldayani, S.T., M.T.

Dosen Pembimbing Anggota : Ir. Khairul Anam, S.T., M.T., Ph.D., IPM

PENGESAHAN

Skripsi yang berjudul "IMPLEMENTASI ALGORITMA YOLO PADA MODUL KAMERA UNTUK DETEKSI JENIS DAN KECEPATAN KENDARAAN" karya Moh. Faizin telah disetujui pada:

Hari : Selasa

Tanggal : 24 Januari 2023

Tempat : Fakultas Teknik Universitas Jember

Tim Penguji,

Ketua,	Anggota I,
Ir. Wahyu Muldayani, S.T., M.T.	Ir. Khairul Anam, S.T., M.T., Ph.D., IPM
NIP. 760016799	NIP. 197804052005011002
Anggota II	Anggota III,
Ir. Widya Cahyadi, S.T., M.T.	Ir. Arizal Mujibtamala Nanda Imron, S.T., M.T.
NIP. 198511102014041001	NIP. 760017099

Mengesahkan,
Dekan Fakultas Teknik Universitas Jember

Dr. Ir. Triwahju Hardianto, S.T., M.T.
NIP. 197008261997021001

RINGKASAN

Implementasi Algoritma YOLO pada Modul Kamera Untuk Deteksi Jenis dan Kecepatan Kendaraan; Moh. Faizin, 181910201105 ; 2023 ; 116 halaman ; Jurusan Teknik Elektro, Fakultas Teknik, Universitas Jember.

Kemajuan teknologi di dunia yang semakin berkembang dengan sangat pesat. Satu diantaranya yakni perkembangan teknologi kecerdasan buatan sering disebut juga dengan AI (*Artificial Intelligence*). AI merupakan suatu simulasi manusia yang dimodelkan kedalam sebuah mesin, yang kemudian mesin tersebut diprogram sehingga dapat berpikir layaknya manusia. Salah satu contoh implementasi AI yaitu sebagai penghitung kendaraan, kecepatan kendaraan dan jenis kendaraan yang melintas di jalan raya. Hal ini dapat membantu pihak kepolisian dalam menertibkan lalu lintas dan mencegah terjadinya kecelakaan yang disebabkan oleh masyarakat baik disengaja maupun tidak. Salah satu tujuan dari penelitian ini yaitu untuk membuat sistem pemantauan kecepatan kendaraan berdasarkan kelas (mobil dan motor) yang melintas dengan menggunakan algoritma YOLOV5 yang akan digunakan untuk mendeteksi kendaraan. Pembuatan sistem ini menggunakan dataset sebanyak 300 gambar yang didapat dari rekaman CCTV yang di konversi ke gambar berdasarkan waktu yang berbeda-beda (pagi, siang, sore dan malam). Dataset tersebut akan dilatih dengan menggunakan hyperparameter berbeda yang terdiri dari Epoch, Batch size, Optimizer dan model YOLOV5 yang digunakan. Pengujian ini didapatkan hasil data yang baik pada nilai Epoch 300, Batch 8, Optimizer *Stochastic Gradient Descent* (SGD) dan model YOLOV5s. Hasil data tersebut menghasilkan nilai mAP sebesar 0,97355. Hasil data tersebut dipilih berdasarkan referensi jurnal dan pengujian ANOVA yang telah dilakukan. Selanjutnya pengujian intensitas cahaya, Pengujian ini bertujuan untuk mengetahui pengaruh dari intensitas cahaya terhadap objek yang akan dideteksi. Kamera yang digunakan berukuran (1920x1080). Sistem dapat mendeteksi kendaraan dengan baik pada rentang nilai intensitas cahaya antara 7000 lux hingga 15000 lux dengan menghasilkan rata-rata error persen sebesar 17,39% untuk kelas mobil dan 8,19% untuk kelas motor. Intensitas cahaya pada nilai 205 lux yang diambil pada malam hari dihasilkan deteksi kendaraan dari sistem yang tidak akurat. Pengujian *real time*

yang dilakukan pada siang hari dengan pengambilan data sebanyak 10 objek pada rentang nilai intensitas cahaya yang telah di uji sebelumnya. Pengujian dengan hasil error persen terendah dan terbesar diperoleh sebesar 1,28% dan 12,97%.

SUMMARY

***Implementation of the YOLO Algorithm on the Camera Module for Vehicle Type and Speed Detection**; Moh. Faizin ; 181910201105 ; 2023 ; 116 pages ; Department of Electrical Engineering, Faculty of Engineering, University of Jember.*

Technological progress in the world is developing very rapidly. One of them is the development of artificial intelligence technology or often referred to as AI. AI (Artificial Intelligence) is a human simulation that is modeled into a machine, which is then programmed, so that the machine can think like a human. One example of applying AI is a vehicle counter, vehicle speed and type of vehicle crossing the highway. This can help the police in regulating traffic and preventing accidents caused by the community, whether intentional or unintentional. One of the goals of this research is to create a vehicle speed monitoring system based on passing classes (cars and motorbikes) using the YOLO algorithm to detect vehicles. The creation of this system uses a dataset of 300 images obtained from CCTV recordings which are converted into images based on different times (morning, afternoon, evening and night). The dataset will be trained to use different hyperparameters consisting of Epoch, Batch size, Optimizer and the YOLOV5 model used. This test obtained good data results on the Epoch 300, Batch 8, Optimizer Stochastic Gradient Descent (SGD) and the YOLOV5s model. The data results produce a mAP value of 0.97355. The results of the data were selected based on journal references and the ANOVA test that had been carried out. Next, a light intensity test is carried out on the object to be detected. The camera used is (1920x1080). The system can detect vehicles properly in the range of light intensity values between 7000 lux to 15000 lux by producing an average percent error of 17.39% for the car class and 8.19% for the motorcycle class. Light intensity with a value of 205 lux taken at night resulted in inaccurate detection of vehicles from the system. Real time testing is carried out during the day by collecting data on 10 objects in the range of light intensity values that have been tested before. Real time testing with the lowest and largest percent error results obtained at 1.28% and 12.97%.

PRAKATA

Puji syukur kehadirat Allah SWT karena atas ni'mat, karunia serta hidayah-Nya yang telah memberikan kemudahan bagi penulis dalam menyelesaikan penelitian ini dengan baik. Sholawat dan salam selalu tercurahkan kepada junjungan Nabi besar Muhammad SAW sebagai suri tauladan dan rahmat bagi seluruh alam semesta. Dengan segala kerendahan hati, penulis mengucapkan terimakasih banyak kepada:

1. Allah SWT yang Maha Pengasih dan Maha Penyayang;
2. Nabi Muhammad SAW, sebagai suri tauladan bagi seluruh umat;
3. Kedua orang tua saya, Bapak Moh. Yasin dan Ibu Ruqiah yang telah membesarkan, mendoakan dan mendidik saya hingga menjadi seperti saat ini, serta adik saya Rhobiatul Adawia;
4. Dosen pembimbing, Bapak Ir. Wahyu Muldayani, S.T., M.T. dan Bapak Ir. Khairul Anam, S.T., M.T., Ph.D., IPM yang telah memberikan arahan dan bimbingannya selama pengerjaan skripsi ini;
5. Dosen penguji, Bapak Ir. Widya Cahyadi, S.T., M.T. dan Bapak Ir. Arizal Mujibtamala Nanda Imron, S.T., M.T. yang telah memberikan kritik dan saran kepada penulis;
6. Bapak Widhi Winata Sakti, S.T., M.T. yang telah menyalurkan ilmunya dan membimbing penulis dalam pengerjaan alat pada skripsi ini;
7. Adik perempuan Rhobiatul Adawia yang selalu menghibur penulis saat proses pengerjaan skripsi;
8. Seluruh guru TK Al-Hidayah VI, SDN 1 Wringin Anom, SMPN 5 Situbondo, MAN 2 Situbondo dan Seluruh dosen Teknik Elektro Universitas Jember atas ilmu yang telah diberikan kepada saya selama ini;
9. Keluarga besar Laboratorium Elektronika dan Terapan Fakultas Teknik, Lembaga Pengabdian Masyarakat, dan Laboratorium Sistem Cerdas dan Robotika-CDAST Universitas Jember yang telah menyediakan tempat dalam pembuatan alat pada skripsi ini;

10. Keluarga besar Teknik Elektro Universitas Jember Angkatan 2018 dan Almamater tercinta;
11. Partner saya, Waki'atil Rosida, S.Si, yang selalu memotivasi penulis.
12. Sahabat saya, Vina Alvi Varhanah, Nailah Altoffina, Fitria Atiqatul Fauziyah, Apt. Siti Zulaiha, S. Farm, Apt. Tasya Salsabila Multazam, S. Farm dan Firman Fathor Rahman Shodiq, S.M. yang selalu memotivasi dan menyemangati penulis dalam mengerjakan Skripsi ini;
13. Teman-teman seperjuangan Fikri, Alfasindo Bagus E., S.T. , Kholis Anwar F., S.T. , Andi Kurniawan, S.T., Ilham, Ikbal Sari Sakti, S. Farm, Faizal dan teman-teman lainnya yang selalu memotivasi penulis untuk mengerjakan skripsi ini, serta teman-teman seperjuangan di Laboratorium Elektronika Terapan, Laboratorium Sistem Cerdas dan Robotika CDAST Universitas Jember;
14. Seluruh pihak yang terlibat dan tidak bisa penulis sebutkan satu persatu.

Harapan penulis dengan terbitnya karya tulis ini akan memunculkan berbagai kritik dan saran yang akan membangun terbentuknya sistem yang lebih baik lagi. Semoga karya tulis ini dapat bermanfaat bagi semua pihak.

Jember, 24 Januari 2023

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Kemajuan teknologi di dunia yang terjadi saat ini semakin berkembang dengan sangat cepat. Satu diantaranya yakni perkembangan teknologi kecerdasan buatan atau yang sering disebut juga dengan *Artificial Intelligence*. *Artificial Intelligence* (AI) merupakan simulasi manusia yang dimodelkan kedalam sebuah mesin yang kemudian mesin tersebut diprogram sehingga dapat berfikir layaknya manusia pada umumnya. Kecerdasan buatan tidak hanya sebatas pada sesuatu yang memiliki bentuk fisik saja, seperti halnya sebuah robot. Kecerdasan buatan yang sebenarnya adalah suatu program yang memiliki bentuk matematis atau instruksi, berbeda dengan program pada umumnya yang menghasilkan aksi berdasarkan instruksi. Hingga saat ini kecerdasan buatan (*Artificial Intelligence*) sudah mencakup berbagai macam bidang mulai dari yang umum (pembelajaran dan persepsi) sampai yang khusus misalnya untuk mendiagnosis penyakit dan lain sebagainya (Davis dkk., 2010).

Salah satu implementasi dari kecerdasan buatan (*Artificial Intelligence*) saat ini yaitu penggunaan AI sebagai penghitung kendaraan, kecepatan kendaraan dan jenis kendaraan yang melintas di jalan raya. Pemantauan terhadap kondisi lalu lintas sangatlah penting dengan bertambahnya jumlah penduduk maka akan mempengaruhi banyaknya kendaraan di Indonesia. Sebagai upaya untuk mencegah terjadinya kecelakaan lalu lintas yang disebabkan oleh masyarakat baik sengaja maupun tidak di sengaja melakukan pelanggaran lalu lintas di jalan raya, maka parameter pemantauan yang harus dilakukan salah satunya yaitu melakukan pengawasan terhadap laju atau kecepatan kendaraan yang melewati jalan tersebut. Sistem pemantauan yang biasa digunakan saat ini kebanyakan dilakukan secara manual dengan menempatkan beberapa petugas kepolisian di beberapa titik arus lalu lintas yang sering terjadi pelanggaran lalu lintas. Pemantauan dengan sistem yang demikian dianggap masih kurang efektif karena tidak selamanya pihak

kepolisian bisa setiap saat berada di titik pengawasan. Mengatasi masalah yang demikian, perlu adanya suatu sistem yang dapat membantu pihak kepolisian dalam melakukan pemantauan yang dapat mengawasi kondisi arus lalu lintas setiap saat meskipun pihak kepolisian yang ditugaskan sedang tidak berada di titik pengawasan. Memanfaatkan perkembangan teknologi kecerdasan buatan, masalah terkait pemantauan terhadap kecepatan kendaraan dapat diatasi dengan menggunakan Kecerdasan buatan (AI) dengan menggunakan bantuan Kamera CCTV (*Closed-circuit Television*) yang terpasang di setiap sudut jalan yang ada di kota-kota besar khususnya di Kabupaten Jember. Penggunaan image processing yang tertanam di dalam sebuah kamera merupakan suatu hal yang cocok untuk memproses pengolahan citra digital.

Penelitian yang sering bermunculan pada tahun-tahun belakangan ini yaitu teknik *image processing* yang diaplikasikan pada kehidupan seperti pengenalan pola, pengenalan bentuk benda dan lain sebagainya serta berbagai metode yang terus bermunculan. Sehingga untuk memperoleh kecepatan kendaraan yang melintas di suatu jalan maka sistem yang akan dibuat tersebut perlu mengenali atau mendeteksi suatu objek kendaraan yang terekam oleh kamera CCTV.

Penelitian terkait yang sebelumnya sudah dilakukan oleh Muhammad Dandi Susanto, Mahasiswa Universitas jember, yang berjudul "Rancang Bangun Modul Kamera Pengukur Kecepatan Kendaraan Menggunakan Metode *Background Subtraction*". Penelitian tersebut menghasilkan tingkat keakuratan dari pembacaan kecepatan kendaraan dengan rata-rata sebesar 93.2% serta objek yang masih dapat terdeteksi pada saat intensitas cahaya dalam jangkauan 1000 sampai 8000 Lux (Susanto, 2020).

Penelitian lain terkait pengukuran kecepatan kendaraan ini juga dilakukan oleh Angga Riyan Fredianto, Mahasiswa Universitas Jember, yang berjudul "Rancang Bangun Modul Kamera Deteksi Kecepatan Kendaraan Menggunakan NVIDIA JETSON NANO Dengan Metode *Image Pixel Manipulation and Calculation*". Penelitian tersebut menghasilkan tingkat keakuratan dalam mendeteksi kecepatan kendaraan yang sangat baik pada sudut pandang kamera 27 derajat dengan garis ROI berjarak 18 m dari kamera. Adapun intensitas cahaya yang

paling baik dalam mendeteksi kecepatan kendaraan pada penelitian ini yaitu pada range 1000 sampai 7000 Lux. Sistem pendeteksi kendaraan ini memiliki terdapat kekurangan dimana ketika kecepatan kendaraan berada diatas kecepatan 80 km/jam, CCTV tidak dapat mengidentifikasinya, serta kendaraan yang diidentifikasi hanya kendaraan roda empat yang melewati area ROI line dengan jarak 18m dari kamera (Feridianto, 2019).

Penelitian lainnya dilakukan oleh Fadlan Raka Satura, Alfan Adi Chandra, dan Faisal Dharma Adhinata yang berjudul "Pengukur Kecepatan Kendaraan Menggunakan Algoritma *Image Subtraction*". Penelitian ini menggunakan metode *image subtraction* dengan gabungan *Gaussian Mixture Model* (GMM), metode ini digunakan untuk mendeteksi suatu objek yang bergerak. Adapun hasil yang diperoleh selama pengambilan data didapatkan beberapa nilai error yang cukup besar yaitu 23,24% pada saat kecepatan kendaraan dengan rata-rata 30 sampai dengan 40 km/jam. Ketika kecepatan kendaraan yang terdeteksi diatas 50 km/jam, maka gambar dari kendaraan tersebut akan disimpan pada penyimpanan personal komputer (Satura dkk., 2021).

Penelitian lainnya menyangkut kendaraan dilakukan oleh Faqih Rofii, Gigih Priyandoko, M. Ifan Fanani, dan Aji Suraji yang berjudul "*Vehicle Counting Accuracy Improvement By Identity Sequences Detection Based on Yolov4 Deep Neural Networks*". Penelitian ini menggunakan metode *Yolov4*. Metode yang dilakukan telah mampu menghitung jumlah kendaraan berupa mobil, sepeda motor, bus dan truk. Pengujian terbaik dari penelitian ini ketika dilakukan pada siang hari dimana posisi kamera berada pada ketinggian 6 m dan sudut 50° sebesar 83%, 93% dan 94%. Sedangkan akurasi terendah didapat ketika pengujian dilakukan pada malam hari dimana posisi kamera berada pada ketinggian 1,5 m dan sudut 90° sebesar 68%, 77% dan 78% (Rofii dkk., 2021).

Berdasarkan penelitian terkait diatas masih terdapat banyak kekurangan pada metode yang digunakan dalam pengukuran kecepatan kendaraan, maka penelitian ini akan menggunakan metode YOLO yang akan digunakan untuk mendeteksi objek kendaraan dengan menggunakan input video yang terekam oleh kamera CCTV. Harapan pada penelitian ini akan mampu memberikan hasil yang lebih baik

dari hasil yang telah dilakukan pada penelitian sebelumnya serta dihasilkan juga error yang lebih kecil dan dapat membantu atau meringankan pekerjaan pihak kepolisian dalam melakukan pengawasan terhadap pelanggaran lalu lintas.

1.2 Rumusan Masalah

Berdasarkan penjabaran latar belakang diatas maka dapat dirumuskan masalah sebagai berikut:

1. Bagaimana cara memanfaatkan kamera CCTV agar dapat membaca *object* yang sedang berjalan?
2. Bagaimana efektivitas penggunaan metode YOLO dalam melakukan pendeteksian suatu objek serta pengukuran kecepatan kendaraan?
3. Bagaimana pengaruh intensitas cahaya terhadap hasil pendeteksian objek dan pengukuran kecepatan kendaraan?

1.3 Batasan Masalah

Adapun batasan masalah pada penelitian tugas akhir ini agar tidak terlalu luas, diantaranya sebagai berikut:

1. Menggunakan NVIDIA Jetson Nano sebagai pemrosesan citra digital.
2. Posisi kamera dalam keadaan diam dengan latar belakang yang tidak berubah-ubah
3. Kendaraan bergerak dalam satu arah
4. Menggunakan Video hasil rekaman dari CCTV yang terpasang di jalanan.
5. Hanya mendeteksi dua buah objek yaitu mobil dan motor.

1.4 Tujuan

Berdasarkan permasalahan diatas tujuan dilakukannya penelitian ini adalah sebagai berikut:

1. Mengetahui cara pembacaan object dengan memanfaatkan input dari rekaman CCTV.
2. Mengetahui efektivitas penggunaan metode Yolo dalam melakukan pendeteksian suatu objek yang bergerak dan pengukuran kecepatannya.

3. Mengetahui pengaruh intensitas cahaya terhadap keakuratan dalam melakukan pendeteksian.

1.5 Manfaat

Manfaat dari adanya penelitian ini diharapkan dapat membantu pihak kepolisian dalam melakukan pemantauan lalu lintas guna meminimalisir angka kecelakaan lalu lintas serta diharapkan dapat menertibkan kendaraan yang melintas di jalan raya.

1.6 Sistematika Penulisan

Secara garis besar penyusunan skripsi ini memiliki sistematika penulisan sebagai berikut:

BAB 1 PENDAHULUAN

Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB 2 TINJAUAN PUSTAKA

Bab ini berisi tentang penjelasan teori yang saling berkaitan dengan penelitian yang akan dilakukan.

BAB 3 METODOLOGI PENELITIAN

Bab ini berisi tentang penjelasan metode yang digunakan dalam proses penyelesaian skripsi.

BAB 4 ANALISIS DAN PEMBAHASAN

Bab ini berisi tentang hasil penelitian yang diperoleh serta analisis dari hasil penelitian.

BAB 5 PENUTUP

Berisi tentang kesimpulan yang telah diperoleh dari hasil penelitian tugas akhir ini serta saran untuk pengembangan lebih lanjut terkait topik penelitian ini.

BAB 2. TINJAUAN PUSTAKA

Bab kedua ini akan menjelaskan beberapa hal terkait teori-teori yang mendukung dalam proses penyelesaian penelitian ini yang berjudul "Implementasi Algoritma YOLO pada Modul Kamera untuk Deteksi Jenis dan Kecepatan Kendaraan". Adapun teori yang akan dibahas pada bab ini diantaranya, bagaimana cara memproses suatu gambar (*image processing*), metode yang digunakan (*Yolo*), citra (*image*) beserta elemen-elemen yang terkandung di dalam sebuah gambar. *Image processing* ini nantinya dapat membantu dalam melakukan pengukuran terhadap kecepatan kendaraan dengan menggunakan rumus yang ada.

2.1 Citra (Gambar)

Citra merupakan suatu gambar yang mempunyai kemiripan dari suatu objek pada bidang dwimatra (dua dimensi). Citra dibagi menjadi 2 kategori, diantaranya yaitu citra analog dan citra digital. Citra analog tidak dapat diproses oleh komputer secara langsung sehingga perlu dikonversi terlebih dahulu menjadi citra digital agar dapat diproses oleh komputer. Gambar yang dihasilkan oleh peralatan-peralatan digital dapat diproses langsung oleh komputer, karena dalam peralatan digital terdapat sebuah sistem sampling dan sistem kuantisasi. Sistem sampling ini merupakan suatu sistem yang dapat mengubah citra analog menjadi citra digital dengan cara membaginya menjadi baris (M) dan kolom (N). Pertemuan antara baris dan kolom disebut dengan Pixel. Nilai M dan nilai N ini sangat mempengaruhi citra digital yang dihasilkan. Semakin besar nilai M dan nilai N maka semakin halus citra digital yang dihasilkan. Sistem kuantisasi adalah salah satu sistem yang terdapat pada peralatan-peralatan digital yang dapat mengubah intensitas analog menjadi intensitas digital atau diskrit, sehingga dapat menghasilkan gradasi warna sesuai dengan kebutuhan. Kedua sistem yang disebutkan diatas (sistem sampling dan sistem kuantisasi) bertugas untuk memotong sebuah citra menjadi M baris dan N

kolom serta menentukan besarnya intensitas sehingga diperoleh resolusi citra yang diinginkan (Andono dkk., 2017).

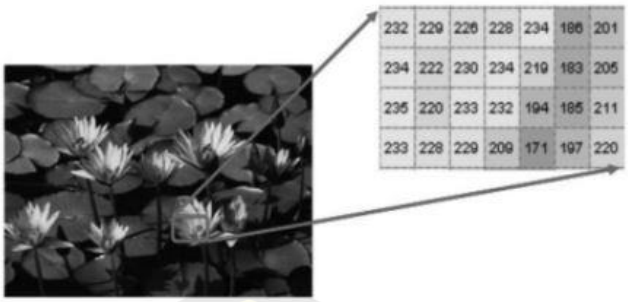
Citra dibedakan menjadi beberapa tipe. Tipe yang sering digunakan didalam penelitian diantaranya yaitu:

1. **Citra Biner (Monokrom)**. Merupakan sebuah citra (gambar) yang tiap pikselnya hanya membutuhkan penyimpanan 1 bit memori. Setiap piksel pada citra biner ini hanya mempunyai dua kemungkinan dari nilai intensitasnya, yaitu 0 atau 1. Gambar 2.1 menunjukkan citra biner dengan nilai intensitas dari setiap pikselnya.

Gambar 2.1 Citra biner dengan nilai piksel 0 atau 1

(Andono dkk., 2017)

2. **Citra Grayscale (Skala Keabuan)**. Sebuah citra yang setiap pikselnya diwakili oleh nilai-nilai intensitas setiap pikselnya dengan kisaran antara 0 sampai 255. Masing masing pikselnya membutuhkan penyimpanan 8 bit memori. Citra 2 bit mewakili 4 warna, 3 bit memori mewakili 8 warna dan seterusnya. Sehingga dapat dikatakan bahwa semakin besar jumlah bit warna yang disediakan maka akan semakin halus gradasi warna yang dihasilkan. Gambar 2.2 merupakan contoh citra grayscale dengan beberapa nilai dari masing-masing pikselnya.



Gambar 2.2 Citra *grayscale* dengan nilai piksel antara 0 sampai 255

(Andono dkk., 2017)

3. **Citra warna (true color)**. Citra yang setiap pikselnya mempunyai kombinasi antara 3 (tiga) komponen warna dasar yaitu merah (*red*), hijau (*green*) dan biru (*blue*) atau yang sering disingkat sebagai RGB. Warna pada setiap piksel ditentukan oleh kombinasi dari intensitas ketiga warna tersebut. Format file grafis menyimpan citra warna sebagai citra 24 bit, yang berasal dari 3 komponen warna (merah, hijau, biru) dengan masing-masing 8 bit. Hal ini dapat menyebabkan citra warna mempunyai 24 juta kemungkinan warna. Gambar 2.3 merupakan citra warna dengan 3 komponen warna dasar (*Red*, *Green*, *Blue*).

Gambar 2.3 Citra warna dengan komponen warna R (merah), G (hijau), B (biru)

(Andono dkk., 2017)

Citra bergerak merupakan suatu gabungan citra yang diam namun ditampikan secara beruntun (sekuensial), sehingga memberikan kesan pada mata manusia sebagai gambar yang bergerak. Gabungan citra tersebut disebut sebagai frame. Jadi

gambar-gambar yang tampak pada suatu film merupakan gabungan dari ratusan frame atau bahkan ribuan frame. (Ardias dkk, 2014)

Berikut merupakan elemen-elemen yang yang terdapat pada sebuah citra digital:

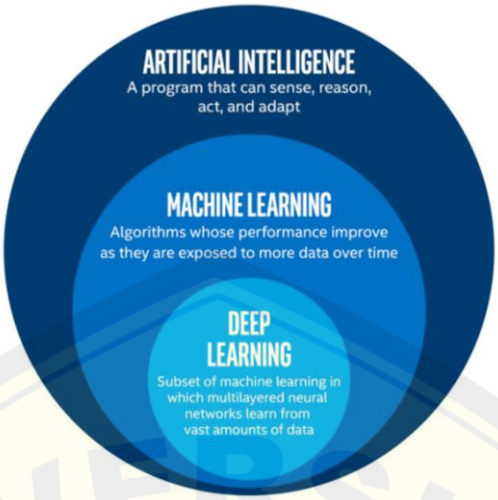
1. Kecerahan (Brightness). Merupakan intensitas cahaya yang dipancarkan oleh suatu piksel dari citra yang dapat ditangkap oleh sistem penglihatan manusia. Kecerahan pada sebuah piksel di dalam sebuah citra merupakan besaran nilai intensitas rata-rata dari area ruang lingkupnya.
2. Kontras (Contrast). Adalah persebaran antara terang dan gelap yang terdapat di dalam sebuah citra. Sebuah citra dikatakan cukup baik apabila sebaran antara gelap dan terang dalam suatu citra cukup merata.
3. Kontur (Contour). Merupakan keadaan yang ditimbulkan dari perubahan nilai intensitas pada piksel-piksel yang berdekatan. Adanya perubahan nilai intensitas ini yang menyebabkan mata manusia mampu mendeteksi tepi-tepi objek di dalam sebuah citra.
4. Warna. Merupakan sebuah persepsi yang ditangkap oleh sistem visual terhadap panjang gelombang cahaya yang dipantulkan oleh sebuah objek

2.2 *Frame Per Second* (FPS)

Frame Per Second (FPS) merupakan banyaknya jumlah gambar yang ditampilkan oleh sebuah layar dalam setiap detik. Sebagai contoh, dalam sebuah film, pada dasarnya film yang ditampilkan di dalam layar tersebut terdiri dari ribuan gambar yang diperlihatkan secara cepat, sehingga dari penglihatan manusia seolah-olah seperti bergerak. Semakin banyak gambar yang ditampilkan pada setiap detiknya dalam membuat gerakan, maka semakin halus kualitas dari gerakan yang dihasilkan (Susanto, 2020).

2.3 Deep Learning

Deep learning adalah bagian dari machine, yang merupakan bagian dari *Artificial Intelligence*. 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*

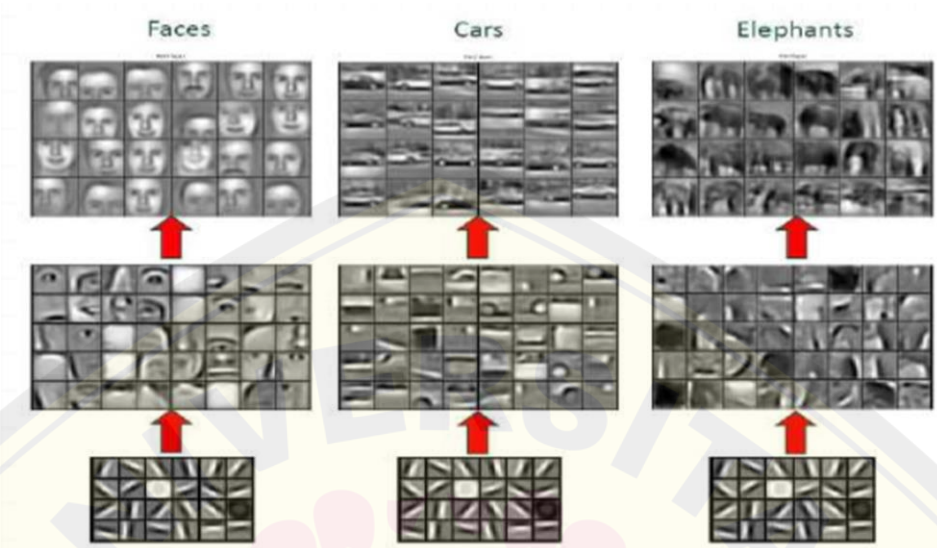
(Hamadi, 2019)

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).

2.4 YOLO (*You Only Look Once*)

YOLO adalah singkatan dari *You Only Look Once* merupakan sebuah algoritma yang pertama kali dikenalkan oleh Joseph Redmon, merupakan basis dari CNN (*Convolutional Neural Networks*) dimana memiliki kemampuan untuk mendeteksi dan klasifikasi objek dalam bentuk gambar ataupun video (Gomaa dkk., 2019). Jaringan saraf konvolusional (CNN), merupakan sebuah metode yang termasuk dalam jenis *Deep Neural Networks* yang arsitektur di dalamnya terdiri dari beberapa jenis *layer* (lapisan) diantaranya *convolutional layer* (Conv), *pooling layer* (*Max pooling*) dan *fully-connected layer* (*Full connection*) dari gabungan lapisan tersebut membentuk suatu arsitektur dari *Convolutional Neural Networks* (CNN). Lapisan awal ini bertugas untuk menangkap fitur tingkat rendah seperti tepi dan sudut, kemudian dilanjut pada lapisan tengah akan menangkap fitur menengah

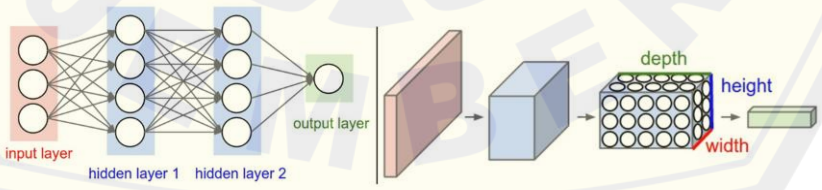
seperti bagian, setelah itu pada lapisan terakhir akan menangkap fitur tingkat tinggi seperti model objek seperti Gambar 2.5.



Gambar 2.5 Representasi pembelajaran *Convolutional Neural Network*

(Hamadi, 2019)

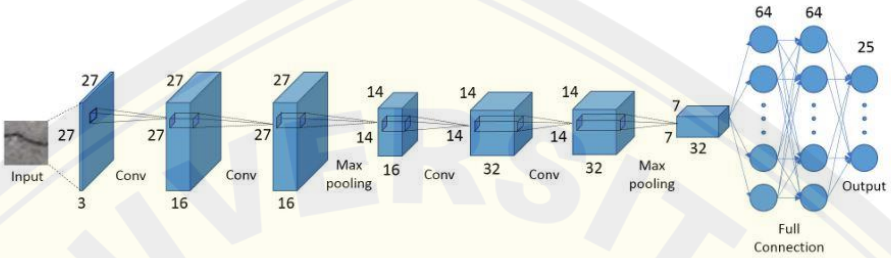
Jaringan *Convolutional Neural Network* terdapat beberapa operasi yang dijalankan di beberapa lapisannya yaitu; *pooling layer* digunakan untuk menghindari *over-fitting* (merupakan suatu keadaan dimana model berusaha untuk mempelajari suatu data secara detail termasuk *noise* yang ada di dalamnya); lapisan klasifikasi digunakan untuk mengklasifikasikan hasil akhir kedalam beberapa kelas. Lapisan-lapisan di *Convolutional Neural Network* terdiri dari 3D volume neuron diantaranya lebar tinggi dan kedalaman. Contohnya seperti pada gambar yang ada di bawah ini.



Gambar 2.6 Lapisan CNN yang mengatur neuronnya di tiga dimensi

(Hamadi, 2019)

Terdapat beberapa arsitektur CNN yang umum digunakan diantaranya LeNet (dikembangkan oleh Yann LeCun yang menggunakan CNN untuk membaca Kode Pos, angka dan lain sebagainya), AlexNet , ZF Net, GoogLeNet, VGGNet dan ResNet. Contoh dari arsitektur CNN yang terdiri dari gabungan *convolutional layer* (Conv), *pooling layer* (Max pooling) dan *fully connected layer* (Full connection). Tumpukan layer tersebut merupakan bagian-bagian penyusun dari arsitektur CNN.



Gambar 2.7 Arsitektur CNN

(Pramestya, 2018)

Berdasarkan arsitektur diatas dibagi menjadi empat bagian, yaitu:

1. *Input layer*. Berdasarkan gambar contoh arsitektur diatas layer akan menyimpan piksel dengan ukuran data 27x27x3 maksudnya 27 merupakan ukuran piksel dari gambar input, sedangkan 3 merupakan channelnya (Red, Green, Blue)
2. *Convolutional layer*. Tahap ini melakukan operasi konvolusi dengan menggunakan sebuah kernel dengan ukuran tertentu. Layer ini akan mempengaruhi keluaran neuron yang terhubung ke *input layer* dengan menggunakan perhitungan skalar produk antara bobot dan daerah *input*. Gambar dibawah ini merupakan ilustrasi dari proses konvolusi dimana terdapat penambahan 1 baris nilai nol (0) di sepanjang garis input.

Gambar 2.8 Representasi matematis dari operasi konvolusi

(Hamadi, 2019)

Pada lapisan konvolusi ini menerima gambar sebagai array 3 dimensi seperti pada persamaan yang ada di bawah ini:

$$y_j = b_j + \sum_i K_{ij} * X_i \quad (2.1)$$

Dimana X_i merupakan input layer, i merupakan nomor filter, K_{ij} merupakan sebuah kernel, b_j adalah bias dan * merupakan operasi konvolusi.

Ukuran output dapat dihitung dengan persamaan sebagai berikut:

$$Output\ size = (Input\ width - Filter\ size + 2 \times Padding)/Stride + 1 \quad (2.2)$$

Dimana langkahnya dengan kecepatan geser saat filter bergerak pada suatu waktu, dan padding menambahkan piksel ekstra ke batas input untuk mengontrol ukuran output dan mempertahankan informasi berguna.

Dalam proses konvolusi juga menggunakan fungsi aktivasi yang bertujuan untuk membuang suatu informasi yang tidak dibutuhkan. Terdapat beberapa fungsi aktivasi yang paling umum digunakan diantaranya:

Fungsi Sigmoid:

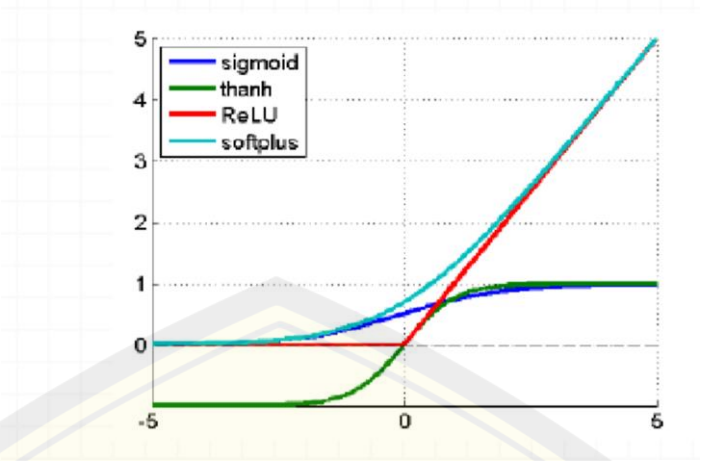
$$sig(x) = 1/(1 + e^{-x}) \quad (2.3)$$

Fungsi hiperbolik:

$$\tanh(x) = (e^{2x} - 1)/(e^{2x} + 1) \quad (2.4)$$

Fungsi Rectified Linear Unit (ReLu):

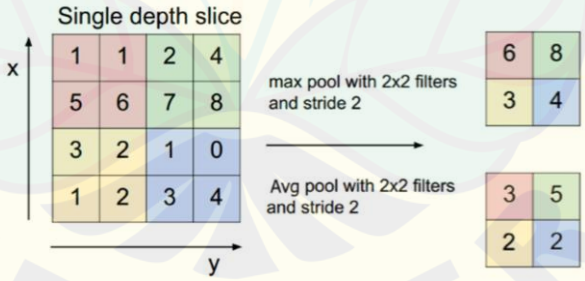
$$f(x) = \max(0, x) \quad (2.5)$$



Gambar 2.9 Fungsi Aktivasi

(Hamadi, 2019)

3. *Pooling layer.* Fungsi utama dari *pooling layer* ini yaitu untuk mengurangi ukuran input tanpa menghilangkan informasi yang terkandung di dalamnya, menyimpan informasi yang relevan dan menghapus informasi yang tidak diperlukan. *Pooling layer* mengoperasikan peta aktivasi ke seluruh input, serta menggunakan fungsi *MAX*. CNN pada umumnya, *max pooling layer* menggunakan kernel dengan dimensi 2x2 dengan artian berpindah sebanyak 2 langkah dalam pergerakan kernelnya. Hal inilah yang menyebabkan ukuran input turun hingga 25% dari semula.

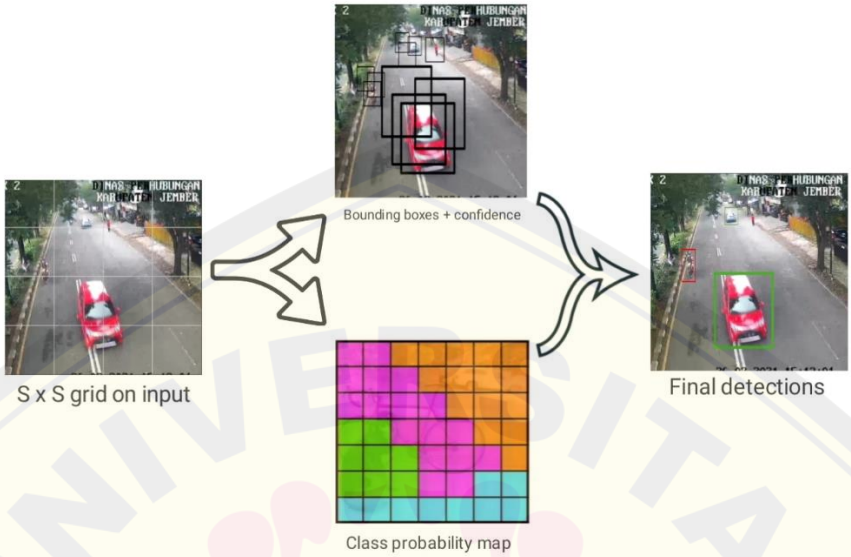


Gambar 2.10 *Max Pooling*

(Hamadi, 2019)

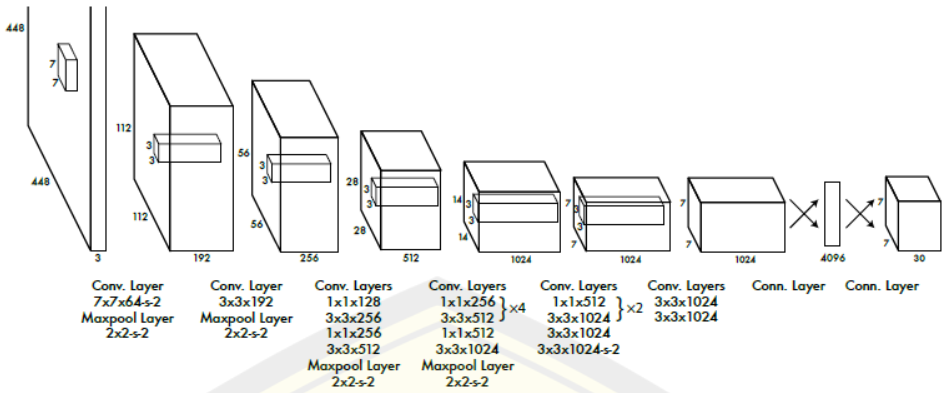
4. *Fully-connected layer.* layer ini berfungsi untuk menggabungkan ekstraksi fitur ke dalam kelas dengan memberikan nilai perkalian acak antara bobot dan bias. Berdasarkan nilai perkalian tersebut akan mempengaruhi pendekatan dari kelasnya. Semakin jauh nilai yang dihasilkan, maka perkalian acak akan

diperbarui kembali dan diulang sehingga membutuhkan waktu yang cukup lama sesuai dengan jumlah fitur yang dihasilkan.



Gambar 2.11 Model YOLO

YOLO bekerja berdasarkan prinsip single shot, maksudnya jaringan arsitektur diatur sedemikian rupa agar dalam satu lintasan bingkai dapat mendeteksi beberapa objek secara serentak. Sistem deteksi YOLO ini menggunakan input berupa gambar atau video dengan menyatukan komponen terpisah ke dalam satu jaringan *neural* 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 SxS grid. Suatu sel grid kecil tersebut yang bertanggung jawab melakukan prediksi jika titik tengah suatu objek jatuh pada grid tersebut. *Bounding box* tersebut mengandung lima elemen prediksi diantaranya *x, y, w, h* dan skor kepercayaan C. koordinat *x* dan *y* mewakili titik tengah dari *box* relatif terhadap batas grid. Elemen *w* (lebar) dan *h* (tinggi) dianggap relatif terhadap keseluruhan gambar (Redmon dkk., 2016).

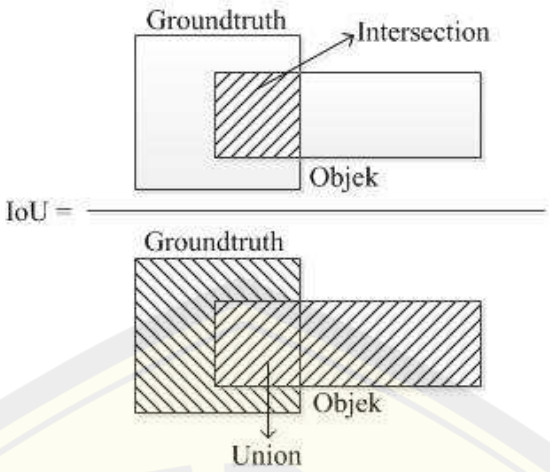


Gambar 2.12 Arsitektur YOLO

(Redmon dkk., 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 $\Pr(Object) * IOU_{pred}^{truth}$. IoU merupakan singkatan dari *Intersection over Union*, merupakan sebuah matrik evaluasi untuk mengukur tingkat keakuratan dalam mendeteksi sebuah objek pada datasheet dengan cara menghitung area pertemuan antara kotak prediksi objek dan kotak kebenaran dasar (*ground truth*) dan membaginya dengan area persatuan mereka. IoU dirumuskan pada persamaan (2.6)

$$IOU = \frac{A \cap B}{A \cup B} \tag{2.6}$$



Gambar 2.13 ilustrasi IoU

(Pramestya, 2018)

Saat objek tidak terdapat pada kotak yang diprediksi maka skor kepercayaan nilainya harus nol. Setiap sel grid juga memprediksi probabilitas kelas bersyarat C, $\Pr(Class_i|Object)$ probabilitas ini digunakan pada sel grid yang berisi objek. YOLO akan memprediksi satu set probabilitas kelas per sel petak, berapapun jumlah kotak B. Saat pengujian akan menggunakan persamaan (2.7) dengan mengalikan probabilitas kelas bersyarat dan prediksi kepercayaan kotak individu (Redmon dkk., 2016).

$$Pr(Class_i|Object) * \Pr(Object) * IOU_{pred}^{truth} = Pr(Class_i) * IOU_{pred}^{truth} \quad (2.7)$$

Berdasarkan persamaan diatas menghasilkan skor kepercayaan khusus untuk masing-masing kotak seperti yang terlihat pada gambar 2.11 Model YOLO.

2.5 Pengukuran Kecepatan Kendaraan

Pengukuran kecepatan kendaraan dapat diperoleh dari perpindahan posisi kendaraan dari setiap framenya. Koordinat titik tengah dari hasil pendeteksian objek yang diperoleh pada setiap frame digunakan untuk mencari nilai perpindahan posisi antar frame. Pendeteksian objek dengan memberikan bounding box yang telah dilakukan sebelumnya harus menentukan posisi awal dan posisi akhir dari objek yang akan dihitung perpindahan jaraknya dengan referensi yang valid. Penggunaan garis deteksi (ROI) ini dapat mempermudah dalam melakukan

perhitungan kecepatan kendaraan, maka pada penelitian yang dilakukan ini akan menggunakan 2 garis deteksi (ROI) pada batas posisi awal dan batas posisi akhir (Susanto, 2020). Persamaan menentukan kecepatan kendaraan dapat dituliskan sebagai berikut.

$$V = {}^{S}/_{t} \tag{2.8}$$

Dimana :

V = kecepatan yang akan dicari

S = jarak sebenarnya antara kedua garis bantu

t = waktu yang diperlukan objek untuk melewati kedua garis bantu

2.6 IP Cam CCTV

IP camera CCTV digunakan pertama kali pada tahun 1996. Istilah "IP Camera" atau "Network Kamera" biasanya digunakan untuk sistem pengawasan keamanan. IP camera merupakan salah satu jenis kamera yang bisa digunakan untuk sistem pemantauan seperti pada toko, kantor, jalan raya serta dapat digunakan untuk mengirim dan menerima data melalui jaringan internet. Adapun keunggulan dari IP camera ini yaitu karena kinerjanya yang cukup kompleks sehingga penggunaan dan konfigurasinya menjadi lebih mudah (Susanto, 2020).

Gambar 2.14 Fisik IP Camera CCTV

(*Hangzhou Hikvision Digital Technology Co.*, 2022)

2.7 NVIDIA Jetson Nano

NVIDIA Jetson Nano merupakan sebuah perangkat yang digunakan untuk menjalankan berbagai muatan *Artificial Intelligence* dengan performa yang cukup tinggi. NVIDIA Jetson Nano Developer Kit mempunyai fitur dan karakteristik khusus dalam menjalankan kerangka kerja AI dan pemodelannya seperti mendeteksi suatu objek, pengenalan suara, pengenalan gambar, segmentasi dan lain sebagainya. Jetson Nano menggunakan sumber 5V 2A oleh micro USB dan dilengkapi dengan pin I/O yang cukup banyak, mulai dari GPIO hingga CSI. Banyaknya pin I/O ini dapat memudahkan *user* untuk menghubungkan berbagai macam sensor untuk keperluan *Artificial Intelligence* (AI). NVIDIA Jetson Nano juga didukung oleh NVIDIA Jetpack, yang meliputi paket dukungan Board (BSP), Linux OS, NVIDIA CUDA, cuDNN, dan TensorRT sebagai *deep learning*, *Computer Vision*, dan komputasi GPU (Feridianto, 2019).

Gambar 2.15 NVIDIA Jetson Nano

(*Corporation*, 2022)

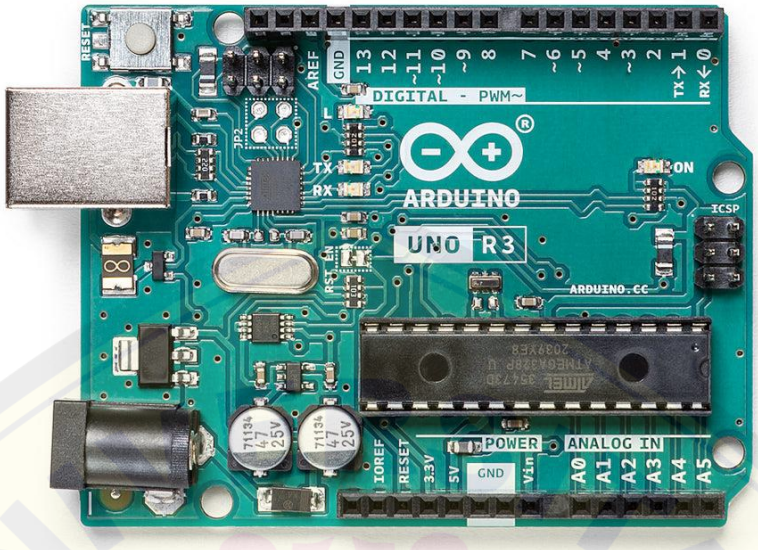
NVIDIA Jetson Nano mempunyai spesifikasi seperti yang ditunjukkan pada Tabel 2.1:

Tabel 2.1 Spesifikasi NVIDIA Jetson Nano

GPU	NVIDIA Maxwell architecture with 128 NVIDIA CUDA® cores
CPU	Quad-core ARM Cortex-A57 MPCore Processor
Memory	4GB 64-bit LPDDR4, 1600MHz 25.6 GB/s
Storage	16GB eMMC 5.1
Video Encode	250MP/sec 1x 4K @ 30 (HEVC) 2x 1080p @ 60 (HEVC) 4x 1080p @ 30 (HEVC) 4x 720p @ 60 (HEVC) 9x 720p @ 30 (HEVC)
Video Decode	500MP/sec 1x 4K @ 60 (HEVC) 2x 4K @ 30 (HEVC) 4x 1080p @ 60 (HEVC) 4x 1080p @ 30 (HEVC) 9x 720p @ 60 (HEVC)
Camera	12 lanes (3x4 or 4x2) MIPI CSI-2 D-PHY 1.1 (1.5 Gb/s per pair)
Connectivity	Gigabit Ethernet, M.2 Key E
Display	HDMI 2.0 and eDP 1.4
USB	4x USB 3.0, USB 2.0 Micro-B
Others	GPIO, I^2C, I^2S, SPI, UART
Mechanical	69.6 mm x 45 mm 260-pin edge connector

(*Corporation*, 2022)

2.8 Arduino UNO



Gambar 2.16 Arduino UNO R3

(SrI-VAT, 2021)

Arduino UNO merupakan sebuah mikrokontroler yang banyak digunakan untuk proyek elektronika yang memanfaatkan mikrokontroler. Arduino UNO juga digunakan sebagai media pembelajaran. Arduino UNO merupakan salah satu mikrokontroler Atmega 328 yang dapat digunakan untuk rangkaian elektronika yang sederhana hingga yang kompleks. Adapun spesifikasi dari Arduino yaitu memiliki *clock speed* yang mencapai 16 MHz, dengan 14 pin I/O digital yang mana 6 pin diantaranya dapat digunakan sebagai output PWM. Arduino juga menyediakan 6 pin untuk input analog dengan resolusi 10 bit (0 sampai 1023). Terdapat beberapa pin I/O arduino yang dapat difungsikan sebagai pin komunikasi serial yang mana salah satunya adalah pin RX dan pin TX yang terletak pada pin D1 dan D0. Arduino UNO memiliki *flash memory* sebesar 32 KB dan EEPROM untuk penyimpanannya. Selain itu, Arduino UNO memiliki fitur lainnya seperti USB *port,* 9V DC *supply port*, header ICSP dan tombol *reset*.

Arduino UNO memiliki perbedaan dari board arduino sebelumnya yaitu tidak menggunakan chip driver FTDI USB-to-serial. Oleh karena itu board tersebut

memberikan fitur yaitu Atmega16U2 (Atmega8U2 hingga versi R2) yang diprogram sebagai USB-to-serial (Farnell, 2013).

Tabel 2.2 *Datasheet* Arduino UNO R3

Microcontroller	ATmega328
Tegangan Operasi	5V
Input Tegangan (Direkomendasikan)	7-12 V
Input Tegangan (Batas Tegangan)	6-20 V
Pin Digital I/O	14 Pin (6 dapat digunakan untuk output PWM)
Pin Analog Input	6 Pin
Arus Pin I/O	20 mA (DC)
Arus DC untuk Pin 3.3V	50 mA (DC)
Flash Memory	32 KB dengan 0.5 KB digunakan untuk *bootloader*
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock speed	16 MHz

(SrI-VAT, 2021)

2.9 Baterai (*Accumulator*)

Baterai merupakan sebuah sumber arus listrik searah atau DC yang berfungsi sebagai mengubah energi kimia menjadi energi listrik. Baterai termasuk dalam golongan elektrokimia, dimana sisi positif baterai menggunakan lempeng oksida dan sisi negatif menggunakan lempeng timbal serta untuk larutan elektrolit dalam baterai menggunakan larutan Asam Sulfat. *Accumulator* (Aki) merupakan sebuah alat yang dapat digunakan untuk menyimpan energi listrik dalam bentuk energi kimia. Pada umumnya aki di Indonesia dianggap sebagai baterai mobil, sedangkan dalam istilah Inggris kata *Accumuluator* dapat mengacu pada baterai, kapasitor dan lain sebagainya (Bidang dkk., 2018).

Aki berfungsi sebagai sumber tegangan penerangan lampu kendaraan pada malam hari ataupun generator listrik yang dilengkapi dengan dinamo starter. Dalam

penelitian ini, aki digunakan sebagai sumber catu daya DC yang digunakan untuk menyalakan Jetson Nano yang digunakan sebagai pengganti PC dalam memproses suatu gambar. Aki yang digunakan merupakan jenis aki kering tipe MF (*Maintenance Free*). Baterai ini merupakan jenis baterai yang bebas dalam perawatannya, serta mempunyai desain khusus yang mampu menekan tingkat penguapan air baterai. Baterai jenis *Maintenance Free* ini biasanya terbuat dari basis jenis baterai *hybrid* maupun baterai kalsium.

Gambar 2.17 Baterai jenis MF (*Maintenance Free*)

(Bidang dkk., 2018)

2.10 Modul XL4016

Modul Step Down XL4016 merupakan sebuah konverter DC to DC yang mempunyai spesifikasi diantaranya tegangan masukan sebesar 4-38V DC, tegangan keluaran sebesar 1.25-36V DC, arus maksimum 8A, daya maksimum sebesar 200W, frekuensi 180KHz dengan suhu kerja kisaran -45 hingga 85 derajat. Modul Step Down ini berfungsi untuk menurunkan tegangan yang stabil sesuai dengan keinginan pengguna dengan mengatur trimpot yang ada pada modul tersebut. Secara umum rangkaian Step Down ini memakai komponen MOSFET (*Metal Oxide Semiconductor Field Effect Transistor*) yang digunakan sebagai switching untuk mengatur duty cycle. Agar mendapat tegangan output yang stabil atau konstan, DC Chopper tipe Buck harus ditambah dengan rangkaian *feedback* sebagai pembanding nilai output dengan nilai referensi. Selisih yang diperoleh dari

tegangan output yang dibandingkan dengan tegangan referensi akan digunakan untuk menghasilkan *duty cycle* PWM yang disesuaikan (*auto adjust*) untuk mengontrol *switching* MOSFET. Semakin banyak selisih yang dihasilkan, maka nilai *duty cycle* juga akan semakin besar (Utami dkk., 2020).

Gambar 2.18 Modul *Step Down* XL4016

(Utami dkk., 2020)

BAB 3. METODOLOGI PENELITIAN

Bab ketiga ini akan menjelaskan terkait beberapa hal pokok terkait tempat dan waktu penelitian, alat dan bahan yang digunakan, tahapan penelitian, diagram alir penelitian, perancangan alat, sistem *processing* data, dan langkah-langkah pengambilan data.

3.1 Tempat dan Waktu Penelitian

3.1.1 Tempat Penelitian

Pelaksanaan penelitian, perancangan, dan pengambilan data alat ini dilakukan di Laboratorium CDAST (*Center for Development of Advance Science and Technology*) Universitas Jember dan di *double way* Universitas Jember.

3.1.2 Waktu Penelitian

waktu penelitian ini dilakukan selama kurang lebih enam (6) bulan dengan rincian penelitian sebagai berikut:

Tabel 3.1 Jadwal Kegiatan Penelitian

No	Kegiatan	Bulan																					
		I				II				III				IV				V				VI			
1	Studi Literatur																						
2	Pembuatan Algoritma																						
3	Pengujian Alat																						
4	Pengambilan Data dan Analisis																						
5	Penulisan Laporan																						

Keterangan

: Pelaksanaan Kegiatan

3.2 Alat dan Bahan

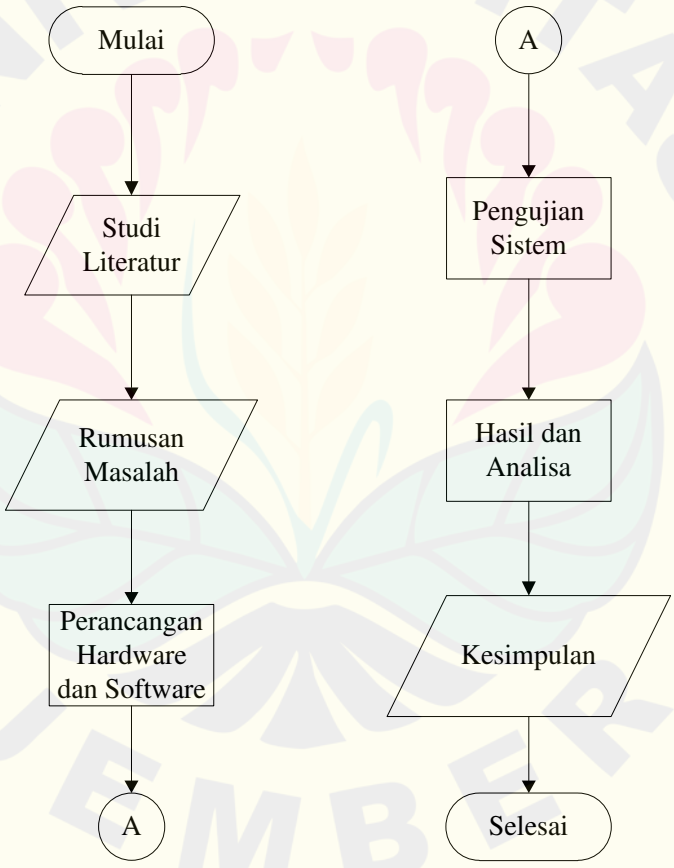
3.2.1 Hardware

1. NVIDIA Jetson Nano
2. Laptop
3. IP Cam CCTV

3.2.2 Software

1. Python
2. Visual Studio Code
3. Video to Jpg Converter
4. LabelImg

3.3 Tahapan Penelitian



Gambar 3.1 Tahapan penelitian

Tahapan penelitian yang dilakukan pada penelitian yang berjudul "Implementasi Algoritma YOLO pada Modul Kamera untuk Deteksi Jenis dan Kecepatan Kendaraan" adalah sebagai berikut:

1. Studi Literatur

   Tahap awal dalam melakukan penelitian ini adalah mencari literatur terkait hasil penelitian yang pernah dilakukan sebelumnya melalui buku, jurnal atau internet guna mengetahui kelebihan dan kekurangan sistem yang dibuat, prinsip kerja, dan teori-teori yang menunjang demi terealisasinya penelitian ini. Literatur yang telah diperoleh diharapkan dapat menjadi masukan agar meminimalisir terjadinya *error* dalam melaksanakan penelitian.

2. Perancangan Alat

   Tahap kedua yaitu perancangan alat, pada tahap ini dilakukan perancangan alat secara sistematis dari alat yang akan dilakukan pada penelitian. Hal-hal yang dilakukan pada tahap ini diantaranya perancangan hardware, perancangan *software* (pembuatan algoritma) serta perancangan sistem monitoring. Diharapkan dari proses perancangan alat yang sistematis ini, alat yang akan dibuat pada penelitian ini dapat terbentuk.

3. Implementasi Alat

   Tahap ketiga, setelah hardware dan software terbentuk maka akan dilakukan pengujian dari sistemnya. Tahap implementasi alat ini akan menguji dari keakuratan sistem dalam pendeteksian jenis-jenis kendaraan serta pengukuran kecepatan kendaraan.

4. Analisa dan Pengambilan Data

   Tahap keempat, setelah melakukan pengujian sistem secara keseluruhan dan memastikan sistem tersebut bekerja dengan baik dan diperoleh hasil yang akurat, maka selanjutnya melakukan pengambilan data yang diperlukan, misalnya pengaruh intensitas cahaya terhadap pendeteksian kendaraan dan pengukuran kecepatannya. Data yang dikumpulkan kemudian dianalisis untuk mengetahui kelebihan dan kekurangan dari penelitian, dengan harapan kekurangan dari penelitian ini dapat disempurnakan oleh penelitian selanjutnya.

5. Penyusunan Laporan

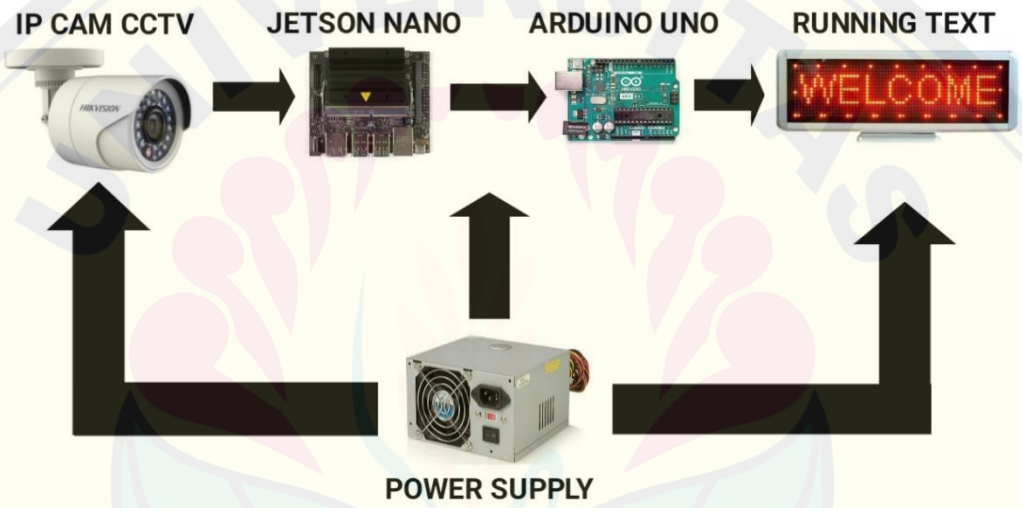
   Tahap akhir yaitu penyusunan laporan. Hasil pengambilan data dan analisa yang telah didapat akan dimasukkan ke pembahasan, kemudian

disimpulkan terkait kinerja dari alat yang telah dibuat. Kekurangan dari penelitian akan dimasukkan ke saran untuk proses perbaikan atau pengembangan pada penelitian yang akan datang.

3.4 Perencanaan Alat

Penelitian yang dilakukan dalam proses perencanaan alat dibagi menjadi beberapa bagian diantaranya; sistem keseluruhan, desain mekanik, deteksi objek kendaraan dengan YOLO. Berikut merupakan penjelasan setiap sistemnya:

3.4.1 Desain Sistem Alat

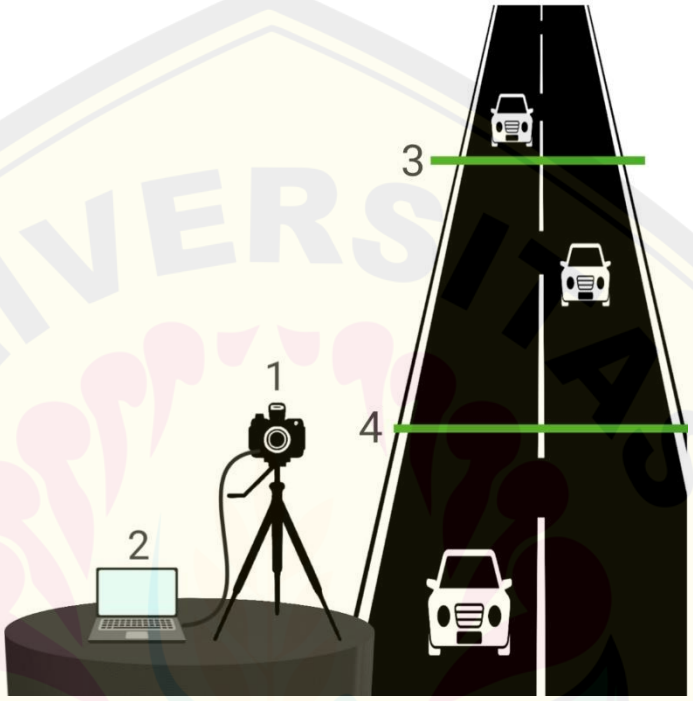


Gambar 3.2 Desain sistem alat

Gambar 3.2 memperlihatkan sebuah input yang diperoleh dari hasil rekaman kamera CCTV. kamera yang digunakan yaitu tipe IP, hal ini karena mempermudah dalam proses mengirim dan menerima data melalui jaringan internet. Data hasil rekaman CCTV kemudian akan diproses menggunakan laptop dengan beberapa tahapan pre-processing seperti merubah video menjadi beberapa frame sehingga menjadi sebuah gambar yang berurutan, proses anotasi dengan pemberian keterangan atau label pada objek berupa kotak pembatas dan kelas objek. Proses anotasi yang dilakukan akan menghasilkan data file dengan format txt. Berdasarkan data yang telah diperoleh melalui tahapan yang dilakukan menggunakan laptop dibagi menjadi 3 proses yaitu *training*, test dan validasi. Hasil yang diperoleh dari

proses menggunakan laptop ini nantinya akan siap diolah oleh NVIDIA Jetson Nano dengan target objek yang dijalankan. Output dari NVIDIA Jetson Nano nantinya akan ditampilkan melalui sebuah monitor.

3.4.2 Desain Sistem Pengambilan Data *Real time*



Gambar 3.3 Desain sistem pengambilan data *real time*

Keterangan:

1. Kamera Webcam
2. Laptop
3. Garis Deteksi/Bantu Pertama
4. Garis Deteksi/Bantu Kedua

Penempatan kamera pada sistem *traffic monitoring* ini nantinya seperti pada Gambar 3.3 diatas, dimana posisi kamera berada pada ketinggian yang tetap. Dua buah garis deteksi (ROI) pada gambar diatas digunakan untuk menghitung kecepatan kendaraan dengan cara menghitung waktu pergerakan frame yang diperlukan objek untuk melewati kedua garis deteksi. Rumus untuk mencari

kecepatan kendaraan dapat dilihat pada Persamaan 2.8 diatas. Nilai jarak diperoleh dari jarak sebenarnya dari kedua garis deteksi (ROI).

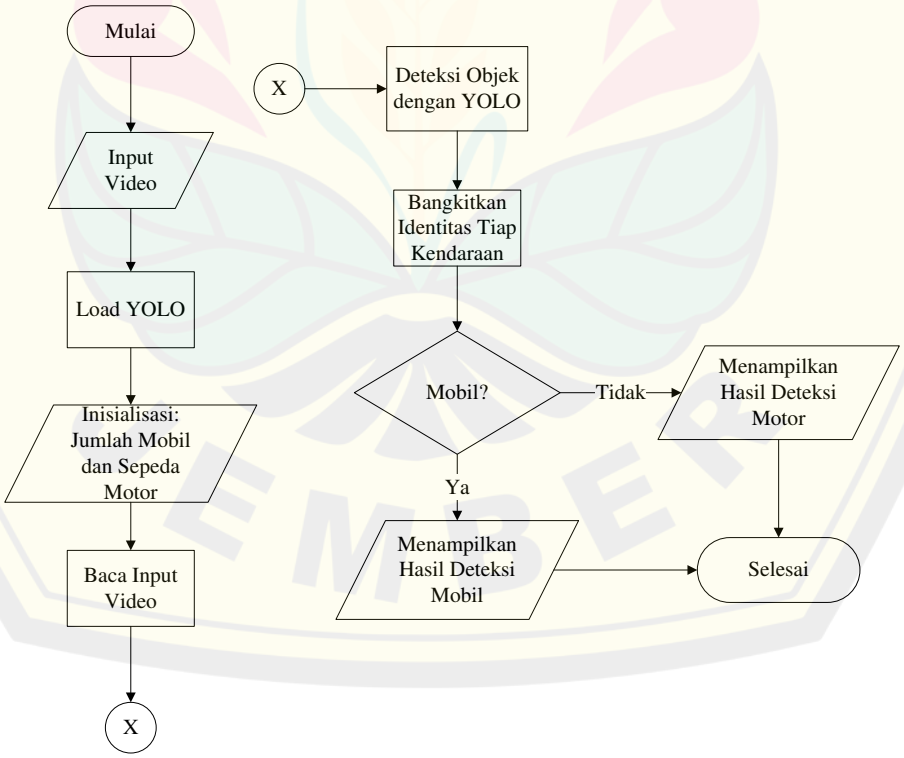
Waktu yang digunakan untuk mencari kecepatan kendaraan diperoleh dari banyaknya frame yang dihasilkan dari objek yang terdeteksi saat melewati garis deteksi pertama sampai garis deteksi kedua. Prediksi kecepatan kendaraan akan muncul ketika objek yang terdeteksi melewati kedua garis deteksi (ROI). Berikut adalah persamaan waktu hasil penurunan dari persamaan FPS (*frame per second*).

$$FPS = Frame / second$$

$$Second = Frame / FPS \quad (3.1)$$

3.4.3 Deteksi Objek Kendaraan dengan YOLO

Mendeteksi suatu objek dengan menggunakan algoritma YOLO (*You Only Look Once*) memiliki beberapa proses. Berikut dibawah ini merupakan diagram alir dari proses deteksi suatu objek dengan YOLO serta penghitungan kecepatan kendaraannya.

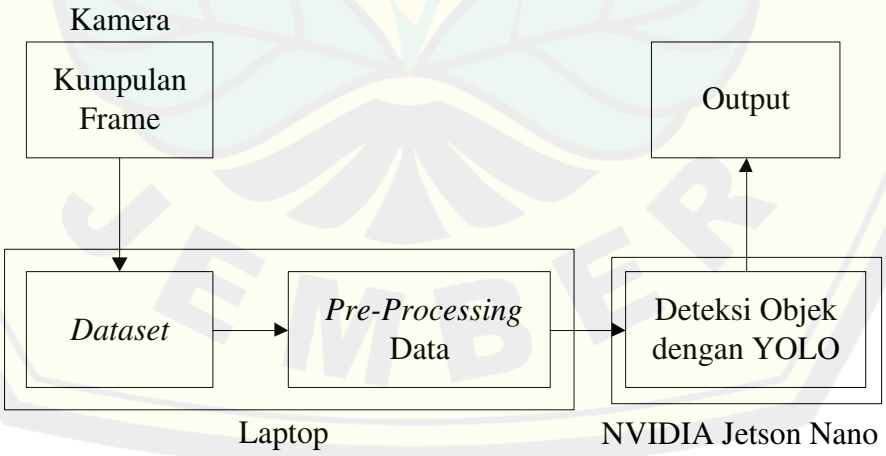


Gambar 3.4 Diagram alir deteksi objek dengan YOLO

Proses dari deteksi objek dengan menggunakan algoritma YOLO ini dimulai dari input yang berupa video dari CCTV, selanjutnya akan masuk pada metode YOLO untuk mendeteksi kendaraan dan jenisnya. Algoritma YOLO ini memuat model dan dataset yang sebelumnya telah disediakan. Tahap selanjutnya inisialisasi jumlah mobil, sepeda motor, dan sebagainya, kemudian dilanjutkan pada proses baca input video. Pembacaan input video ini memuat beberapa proses diantaranya *grayscale, Histogram Equalization, Median Filtering, Gaussian Blur, Otsu Binarization,* dan *Dilatation*. Masuk pada tahap deteksi objek dengan YOLO yang meliputi pemberian *Box*/Kotak, Kelas, dan Skor Kepercayaan., kemudian membangkitkan identitas pada tiap kendaraan. Tahap terakhir pada proses deteksi objek dengan menggunakan algoritma YOLO ini yaitu menampilkan hasil deteksi dan perhitungan kecepatan kendaraan, kemudian menyimpan video output yang dihasilkan.

3.5 Sistem *Processing* Data

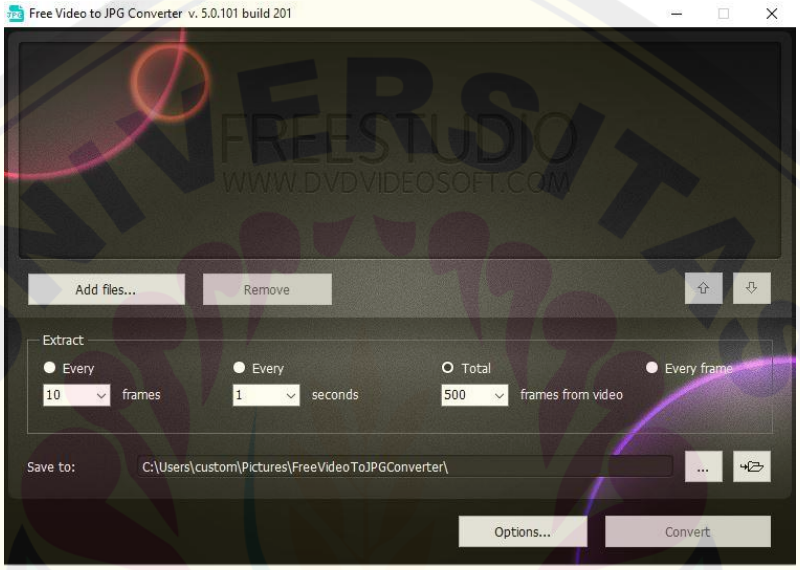
Kumpulan frame yang menjadi sebuah gambar yang berurutan selanjutnya akan melalui beberapa tahapan, mulai dari pengolahan data gambar sampai deteksi objek dengan menggunakan algoritma YOLO. Agar lebih mudah memahami tahapan dalam sistem *processing* data ini, maka dibuatkan diagram blok yang ditunjukkan pada gambar 3.5 dibawah ini.



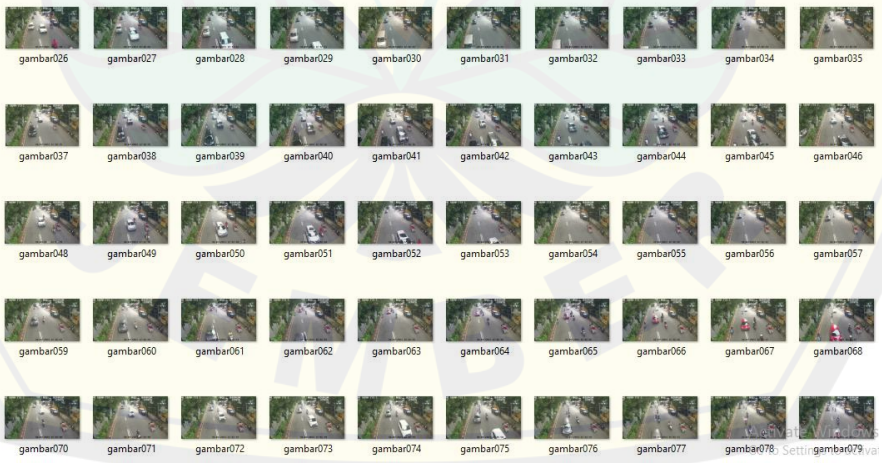
Gambar 3.5 Diagram blok sistem *processing* data

3.5.1 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 CCTV yang di *extract* menjadi beberapa frame sehingga menjadi gambar yang berurutan. Proses *convert* video menjadi sebuah gambar ini menggunakan *software* video *to* JPG *converter.*



Gambar 3.6 Tampilan *Software* Video *to* JPG *Converter*

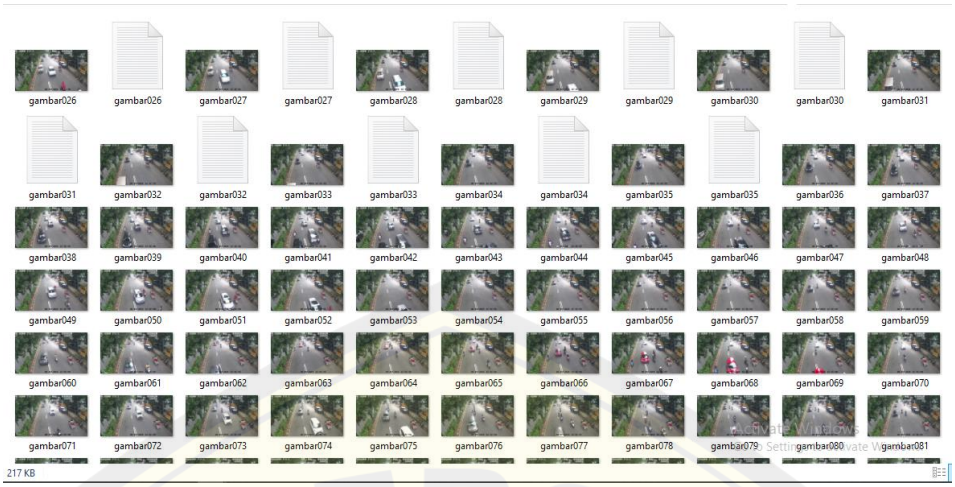


Gambar 3.7 Hasil *Convert* Video *to* JPG sebelum dilakukan pelabelan

3.5.2 Pre-Processing Data

Video yang telah di ekstrak menjadi kumpulan gambar yang berurutan kemudian dilanjut pada tahap *pre-processing* data. Tahap ini terdiri dari dua tahapan 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 *software LabelImg* dapat dilihat pada gambar 3.8 Pemberian label pada objek dibawah ini.

Gambar 3.8 Pemberian label pada objek



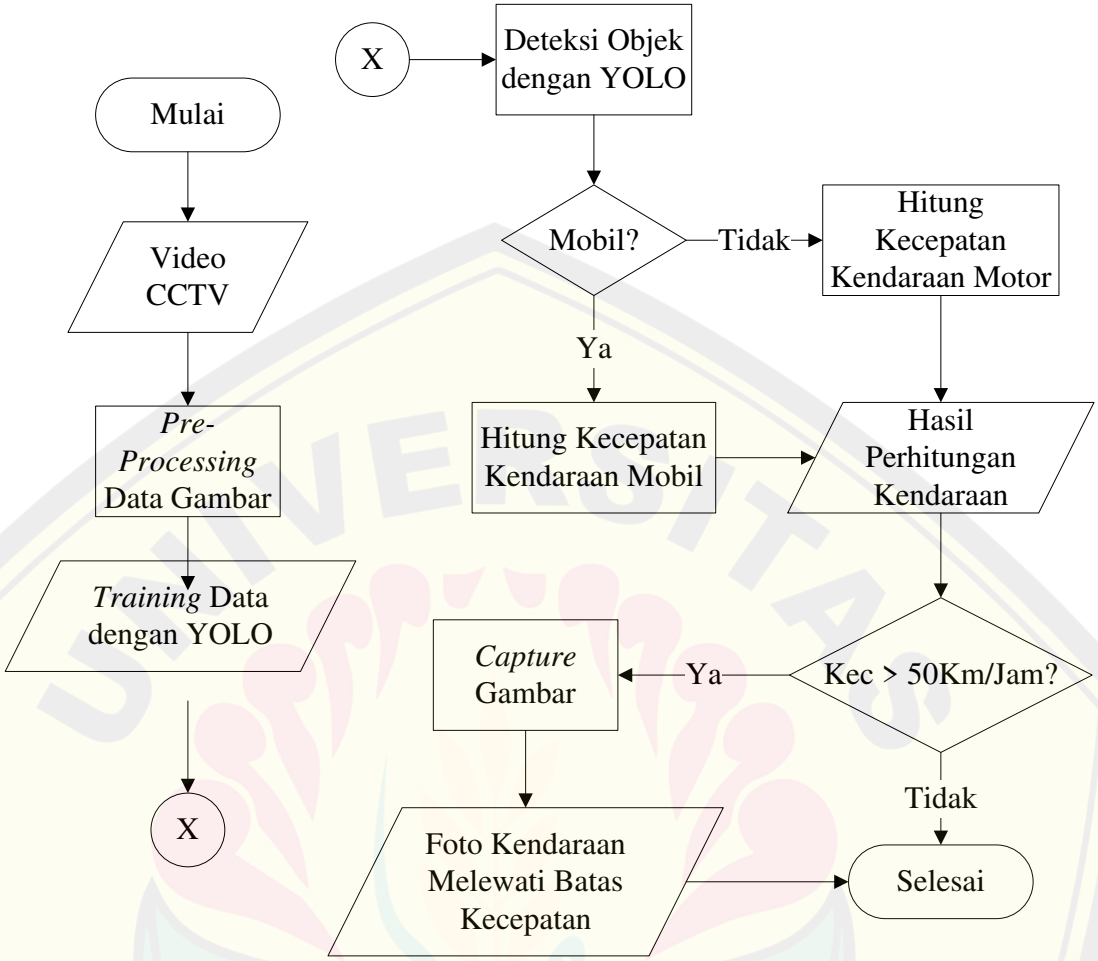
Gambar 3.9 Beberapa *datasheet* telah melalui proses pelabelan

Kelas	X_center	Y_Center	Lebar	Tinggi
1	0.669922	0.884722	0.086719	0.230556
1	0.913672	0.713194	0.055469	0.201389
1	0.363281	0.216667	0.015625	0.050000
1	0.434375	0.142361	0.009375	0.031944
0	0.378516	0.254861	0.039844	0.073611
0	0.354297	0.458333	0.086719	0.202778
0	0.512891	0.520139	0.119531	0.254167
0	0.662109	0.359028	0.116406	0.162500

Gambar 3.10 Contoh hasil file *txt* pada gambar pertama

File *txt* yang dihasilkan dari proses anotasi dapat dilihat pada gambar 3.10 diatas. File tersebut berisi nilai-nilai yang terbagi menjadi 5 bagian. Nilai yang dimulai pada bagian paling kiri merupakan nilai kelas, dilanjut dengan nilai X center, Y center, lebar dan tinggi dari suatu objek yang diberikan kotak pembatas.

3.6 *Flowchart System* Keseluruhan



Gambar 3.11 *Flowchart system* keseluruhan

Berdasarkan *flowchart* sistem diatas alur pembacaannya dimulai dari proses menginputkan data berupa video yang telah di *convert* menjadi beberapa frame kemudian masuk pada tahap *pre-processing* data yang didalamnya mencakup beberapa proses diantaranya *resizing* dan *cropping* serta pemberian anotasi atau kotak pembatas dan sebagainya. *Pre-processing* selesai, dilanjutkan pada tahap *training* data kemudian deteksi objek dengan YOLO. Masuk pada *decision* atau keputusan dimana ketika deteksinya akurat maka akan menampilkan hasil deteksi dan kecepatan kendaraan, sebaliknya jika deteksinya tidak akurat atau deteksi tidak sesuai dengan objek yang dideteksi dan pembacaan kecepatan kendaraan tidak

sesuai maka akan dilakukan proses *training* kembali. Hal ini bertujuan agar diperoleh nilai tingkat akurasi yang maksimal dari suatu sistem.

3.7 Data Pengujian

Pengujian data dilakukan setelah perancangan sistem telah selesai dan bekerja sesuai rancangan. Adapun beberapa hal yang perlu dilakukan pengujian agar diperoleh hasil yang maksimal diantaranya sebagai berikut.

3.7.1 Pengujian Kamera

Pengujian kamera ini bertujuan untuk mengetahui pengaruh dari intensitas cahaya terhadap objek yang akan dideteksi. Kamera yang digunakan berukuran (1920x1080) yang berarti gambar memiliki 1920 piksel secara horizontal dengan 1080 piksel secara vertikal. Pengujian ini rencananya akan dilakukan pengambilan data di waktu yang berbeda diantaranya pagi pada pukul 07.00, siang pukul 12.00, sore pukul 16.00 dan malam hari pada pukul 19.00. Saat pengujian pencahayaan di waktu yang berbeda ini juga dilakukan dengan pengambilan foto dan untuk mengukur intensitas cahayanya menggunakan alat ukur lux meter.

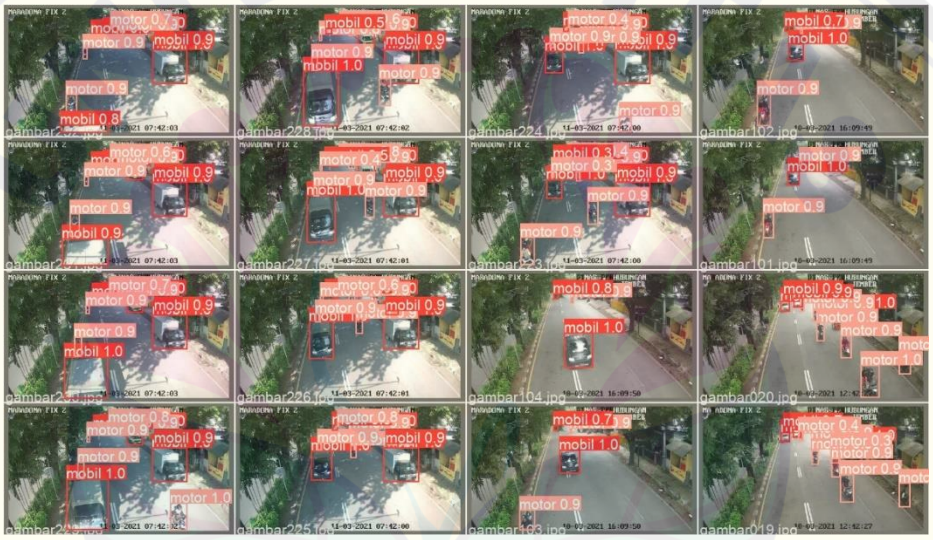
Gambar 3.12 Pengukuran intensitas cahaya dengan lux meter

Gambar 3.12 menunjukkan hasil dari pembacaan lux meter ketika melakukan pengujian pada salah satu waktu yang telah ditentukan. Pengujian dengan

menggunakan lux meter dilakukan dengan cara menyalakan alat ukurnya dan mengarahkan sensornya pada cahaya matahari kemudian dapat dilihat hasilnya pada display. Gambar tersebut diambil pada saat sore hari dengan nilai yang diperoleh sebesar 11.24 Klux atau 11240 lux.

3.7.2 Pengujian Deteksi Jenis Kendaraan

Pengujian deteksi jenis kendaraan ini dilakukan dengan menerapkan algoritma YOLO untuk mendeteksi jenis kendaraan diantaranya mobil, sepeda motor dan sebagainya dari beberapa video yang akan disajikan. Pengujian ini dilakukan untuk mengetahui tingkat akurasi dari algoritma YOLO dalam mendeteksi suatu objek.



Gambar 3. 13 pengujian deteksi jenis kendaraan

Gambar 3.13 menunjukkan hasil dari pengujian deteksi jenis kendaraan. Berdasarkan pengujian tersebut, objek yang terdeteksi akan diberikan sebuah kotak pembatas beserta caption dari objek sesuai dengan kelas yang dideteksi (motor dan mobil). Gambar tersebut terdapat angka di setiap kotak pembatas objek, hal ini menandakan seberapa baik sistem dalam mendeteksi sebuah objek. Angka tersebut merupakan nilai *Confidence* yang dihasilkan dalam pembacaan objek. Ketika nilai *Confidence* semakin mendekati nilai 1.0 maka pembacaan objek tersebut semakin akurat.

3.7.3 Pengujian Pengaruh Intensitas Cahaya terhadap Deteksi Kendaraan

Pengujian pengaruh intensitas cahaya dalam proses deteksi objek kendaraan akan dilakukan dari beberapa video yang akan disajikan. Tingkat keakurasiannya dipengaruhi oleh nilai intensitas cahaya pada saat melakukan proses deteksi objek kendaraan. Oleh karena itu, perlu dilakukan pengujian terhadap pengaruh intensitas cahaya terhadap deteksi jenis kendaraan. Pengujian ini dilakukan setelah melalui tahap pengujian kamera dan pengujian deteksi objek. Hal ini bertujuan untuk mengetahui pengaruh dari intensitas cahaya yang berbeda-beda terhadap hasil deteksi objek oleh sistem. Pengujian ini dilakukan dengan merekam video yang berbeda-beda berdasarkan intensitas cahaya yang telah ditentukan. Video tersebut kemudian digunakan oleh sistem untuk mengetahui seberapa pengaruh dari intensitas cahaya berdasarkan objek yang dideteksi.

3.7.4 Pengujian *Running Text*

Pengujian *running text* ini bertujuan untuk menampilkan hasil deteksi kecepatan kendaraan pada led matrix max7219 yang digunakan. Pengujian ini dilakukan dengan menggunakan program pada arduino yang kemudian dari Jetson Nano dikomunikasikan dengan arduino menggunakan pin komunikasi serial yaitu pin RX dan TX pada arduino. Setelah terhubung, maka Max7219 led matrix display akan menampilkan hasil dari pembacaan serialnya dengan menampilkan angka yang merupakan prediksi kecepatan kendaraan yang terdeteksi oleh sistem.

Gambar 3. 14 Pengujian *running text*

Gambar 3.14 merupakan salah satu hasil yang diperoleh dari beberapa pengujian *running text* yang telah dilakukan. Max7219 ini mempunyai spesifikasi 5 buah pin yang meliputi pin VCC, GND, DIN, CS dan CLK yang mana masing-masing dari pin tersebut dihubungkan pada Arduino UNO dengan pin 5V, GND, pin 11, 3 dan pin 13.

3.7.5 Pengujian Kecepatan kendaraan

Pengujian kecepatan kendaraan kendaraan rencananya akan dilakukan dari beberapa video yang telah disajikan. Perhitungan kecepatan kendaraan ini menggunakan persamaan seperti yang ada di bawah ini.

$$V = {}^{s}/_{t} \tag{3.1}$$

Keterangan:

V = Kecepatan

s = Jarak sebenarnya antara kedua garis deteksi

t = Waktu yang diperlukan suatu objek untuk melewati kedua garis deteksi

Hasil dari deteksi kecepatan kendaraan dengan menggunakan sistem nantinya akan dibandingkan dengan perhitungan kecepatan manual guna mendapatkan tingkat akurasi dari sistem yang telah dibuat dalam mendeteksi kecepatan kendaraan. Setelah melakukan pengujian kecepatan kendaraan, selanjutnya dilakukan pengujian kecepatan kendaraan berdasarkan intensitas tingkat keakurasiannya juga dipengaruhi oleh intensitas cahaya. Oleh karena itu perlu dilakukan pengujian terkait pengaruh intensitas cahaya yang berbeda-beda pada akurasi kecepatan kendaraan.

BAB 4. HASIL DAN PEMBAHASAN

Bab keempat ini akan dijelaskan terkait hasil dari pengujian sistem pendeteksi kecepatan kendaraan menggunakan algoritma YOLOV5 berbasis *Convolutional Neural Network* (CNN) yang telah dirancang. Pembuatan sistem ini dilakukan di laboratorium CDAST (*Center for Development of Advance Science and Technology*) Universitas Jember dengan tujuan dibuatnya sistem ini diharapkan dapat membantu pihak kepolisian Pemerintah Kabupaten Jember dalam melakukan pengawasan, serta memanfaatkan kamera CCTV yang terpasang di sudut jalan untuk mendeteksi kecepatan kendaraan yang melintas dengan penambahan modul NVIDIA Jetson Nano sebagai pemrosesan citra digital.

4.1 Akuisisi Data

Dataset yang dikumpulkan diambil dari rekaman video kamera CCTV yang terpasang di salah satu sudut jalan pemerintah Kabupaten Jember. Rekaman video tersebut kemudian di ekstrak dengan menggunakan *software* Video to JPG *Converter*, 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 300 gambar, dimana dari 300 gambar tersebut terdiri dari waktu yang berbeda-beda diantaranya pagi, siang, sore dan malam hari. Berdasarkan 300 gambar tersebut, dipecah kembali menjadi 2 bagian dengan perbandingan 70% (210 gambar untuk *Training* : 30% (90 gambar untuk Validasi). Alasan dalam penelitian ini mengambil waktu yang berbeda-beda seperti yang disebutkan sebelumnya yaitu agar dapat mengetahui pengaruh intensitas cahaya terhadap sistem dalam mendeteksi sebuah objek, karena seiring berubahnya waktu intensitas cahaya juga akan mengalami perubahan, serta cuaca mendung juga akan akan mempengaruhi besarnya kecilnya intensitas cahaya. Berikut dibawah ini merupakan gambar dari masing-masing waktu yang telah digunakan sebagai dataset:

Gambar 4.1 CCTV pada Pagi Hari

Gambar 4.2 CCTV pada Siang Hari

Gambar 4.3 CCTV pada Sore Hari

Gambar 4.4 CCTV pada Malam Hari

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 dua kelas yaitu mobil dan motor. 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.

4.2 *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 collab atau google collaboratory. Google collab merupakan executable dokumen yang dapat digunakan untuk menulis, menyimpan serta membagikan suatu program yang telah ditulis melalui google drive. Google collab merupakan tempat yang cocok bagi programer yang ingi mengasah pengetahuan mengenai bahasa pemrograman python. Selain gratis, google collab ini dapat dijalankan menggunakan google chrome, mozilla dan sebagainya. Google collab 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 dan motor, 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:

$$precision = \frac{TP}{TP+FP} \tag{4.1}$$

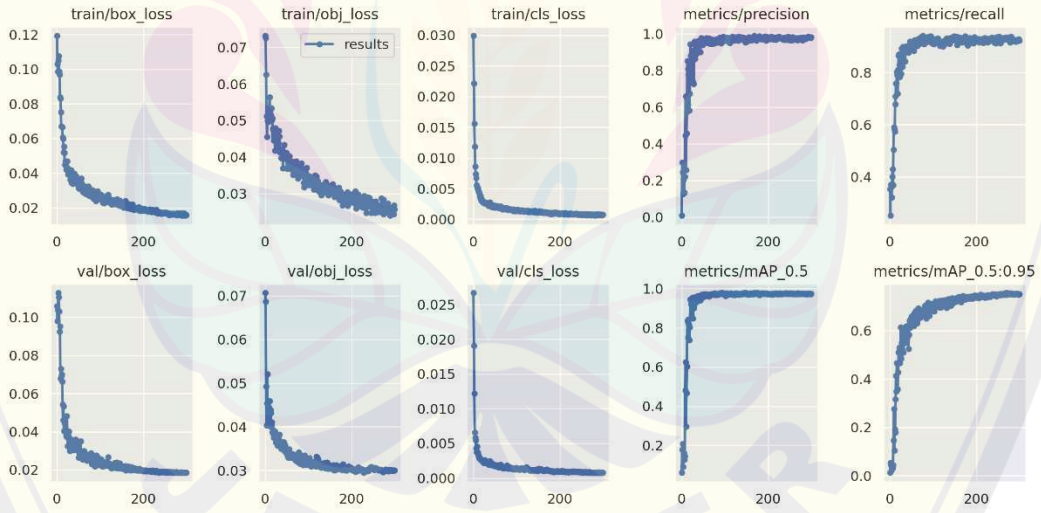
True Positive (TP) merupakan hasil yang diperoleh dari prediksi mesin yang menyatakan benar dan hal tersebut merupakan jawaban yang benar (*true*). *False Positive* (FP) merupakan hasil yang diperoleh berdasarkan prediksi mesin dinyatakan benar namun hal tersebut merupakan jawaban yang salah. Recall

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:

$$Recall = \frac{TP}{TP+FN} \tag{4.2}$$

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 dan motor diperoleh nilai mAP keseluruhan di atas 95%. Berikut merupakan grafik dari kinerja YOLOV5 saat proses *training* ditunjukkan pada gambar 4.5.



Gambar 4.5 Plot *Box*, *Objectness*, *Classification*, *Precision*, *Recall* setiap periode untuk *Training* dan Validasi Dataset

Gambar 4.5 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.

4.3 Hasil Pengujian *Offline* Algoritma YOLO

Pengujian offline dihasilkan dari beberapa proses *training* yang dilakukan dengan menggunakan Google Collab. Tujuan dari pengujian offline ini yaitu untuk mengetahui kinerja dari algoritma YOLOV5 dalam memprediksi suatu objek. Pengujian offline ini akan dilakukan beberapa kali dengan mengubah beberapa parameter atau variabel bebas. Berdasarkan pengujian tersebut nantinya akan dipilih berdasarkan hasil akurasi terbaik sehingga model yang diperoleh akan digunakan atau diuji coba pada pengujian *real time*. Adapun hyperparameter yang dirubah diantaranya yaitu Optimizer, Epoch, nilai Batch, serta model dari YOLOV5.

4.3.1 Pengujian Nilai Epoch

Epoch merupakan suatu parameter yang digunakan untuk menentukan jumlah berapa kali sistem melakukan pengulangan proses pelatihan (*training*) yang akan dijalankan dengan menggunakan dataset yang telah terkumpul sebelumnya pada Google Collab. Epoch merupakan kondisi yang melalui pelatihan yang telah di update dalam satu siklus penuh. Penelitian yang dilakukan ini menggunakan beberapa pengujian dengan mengubah nilai Epoch dalam proses *training* diantaranya yaitu 100 dan 300. Pengujian ini bertujuan untuk mengetahui pengaruh nilai Epoch terhadap mAP yang dihasilkan. Berikut merupakan pengujian dari nilai Epoch dapat dilihat pada tabel 4.1 Hasil pengujian nilai Epoch dibawah ini:

Tabel 4.1 Hasil pengujian nilai Epoch

NO	IMG	EPOCH	BATCH	MODEL	OPTIMIZER	mAP
1	640	100	8	YOLOV5s	SGD	0,97038
2	640	300	8	YOLOV5s	SGD	0,97355

Berdasarkan dari hasil pengujian nilai epoch pada tabel 4.1 diatas dapat dilihat bahwasanya pengujian epoch dilakukan sebanyak 2 kali dengan variasi nilai Epoch yang digunakan yaitu 100 dan 300 dengan parameter yang berbeda-beda. Nilai epoch merupakan faktor yang dapat mempengaruhi hasil akurasi terbaik dari sistem. Hasil terbaik yang diperoleh ketika pengujian epoch ini yaitu pada percobaan kedua dengan nilai epoch sebesar 300. Hal ini karena semakin besar nilai epoch, maka hasil yang diperoleh akan semakin baik. Percobaan pertama, hasil mAP yang diperoleh tidak jauh berbeda dengan percobaan kedua. Hal ini kemungkinan besar dipengaruhi oleh perbedaannya nilai Epoch yang digunakan. namun terdapat kekurangan apabila jumlah Epoch yang digunakan semakin banyak maka proses *training* data akan semakin lama.

4.3.2 Pengujian Nilai Batch

Pengujian Batch *size* mengacu pada besarnya ukuran Batch (jumlah dari pembelajaran mesin atau pelatihan mesin yang digunakan dalam 1 iterasi). Pengujian nilai Batch juga berpengaruh terhadap hasil yang diperoleh oleh sistem. Penelitian ini menggunakan dataset sebanyak 300 frame atau gambar, pengujian Batch yang diuji yaitu dengan ukuran 8 Batch dan 16 Batch. Maksud dari 8 Batch dan 16 Batch yaitu untuk pengujian dengan 8 Batch dari dataset sebanyak 300 gambar tersebut diambil kelipatan 8 frame hingga dataset terakhir. Sampel pertama yaitu gambar ke-1 hingga ke-8 diambil kemudian dilatih atau di *training*. Setelah itu dilanjutkan pada sampel kedua data ke-9 hingga data ke-16 kemudian di *training*, begitu seterusnya. Sama halnya dengan pengujian 16 Batch, dilakukan sampai data terakhir. Berikut merupakan pengujian variasi nilai Batch yang dilakukan pada penelitian ini dapat dilihat pada tabel 4.2 Hasil pengujian nilai Batch dibawah ini:

Tabel 4.2 Hasil pengujian nilai Batch

NO	IMG	EPOCH	BATCH	MODEL	OPTIMIZER	mAP
1	640	300	16	YOLOV5s	SGD	0,97353
2	640	300	8	YOLOV5s	SGD	0,97355

Pengujian Batch yang dihasilkan dapat dilihat pada tabel 4.2 diatas. Seperti halnya pengujian epoch, pengujian nilai Batch juga dilakukan dengan 2 variasi pengujian nilai yaitu 8 Batch dan 16 Batch. Nilai Batch mempengaruhi dari data yang akan diuji. Apabila dataset yang diuji dengan menggunakan Batch yang nilai kelipatannya mendekati banyaknya dataset, maka dataset tersebut hampir semua diuji tanpa adanya data yang terlewatkan. Hasil pengujian terbaik yang diperoleh dari pengujian nilai Batch yaitu pada percobaan kedua pada saat menggunakan 8 Batch diperoleh nilai mAP sebesar 0,97355. Berdasarkan hasil yang diperoleh dapat disimpulkan bahwa nilai mAP mengalami perubahan yang cukup kecil ketika nilai Batch divariasikan, hal ini juga dipengaruhi oleh banyaknya data yang digunakan.

4.3.3 Pengujian Optimizer

Optimasi merupakan salah satu metode yang digunakan dalam *training* jaringan syaraf tiruan. Adapun fungsi dari optimasi yaitu untuk memperoleh nilai bobot yang dapat memberikan keluaran nilai terbaik. Metode optimasi *Gradient Descent* bekerja langkah demi langkah dengan memperkecil suatu nilai fungsi dengan mengubah nilai parameternya. Penelitian yang dilakukan dalam proses pembuatan sistem arsitektur YOLO dalam menentukan klasifikasi objek dilakukan percobaan dengan menggunakan dua buah optimasi atau optimizer diantaranya *Stochastic Gradient Descent* (SGD) dan *Adaptive moment estimation* (Adam).

Optimizer *Stochastic Gradient Descent* (SGD) merupakan suatu teknik pembelajaran suatu sistem yang melakukan update untuk setiap satu data. *Stochastic Gradient Descent* mempunyai konsep yang mirip dengan Batch dengan membagi dataset ke beberapa Batch. Sehingga SGD akan terus mengupdate weight tanpa menunggu selesainya 1 epoch pada saat proses *training*.

Optimizer *Adaptive moment estimation* (Adam) merupakan suatu algoritma optimasi yang prinsip kerjanya menggunakan kombinasi dari algoritma *gradient*

adaptive (AdaGrad) dan *propagation Root Mean Square* (RMSProb) untuk memperbarui bobot berdasarkan data *training*. Algoritma AdaGrad pada optimizer Adam ini akan mempertahankan learning rate dengan menangani masalah atau noise dengan gradient yang menyebar. Sedangkan RMSProp akan mempertahankan learning rate berdasarkan besar gradient terbaru. Berikut merupakan hasil dari pengujian optimizer pada pengujian ini pada tabel 4.3 Hasil pengujian optimizer.

Tabel 4.3 Hasil pengujian optimizer

NO	IMG	EPOCH	BATCH	MODEL	OPTIMIZER	mAP
1	640	300	8	YOLOV5s	Adam	0,97073
2	640	300	8	YOLOV5s	SGD	0,97355

Pengujian optimizer yang dilakukan yaitu dengan 2 jenis optimizer yaitu optimizer SGD dan optimizer Adam. Pengujian terbaik dari pengujian yang telah dilakukan dilihat dari nilai mAP yang diperoleh yaitu pada percobaan kedua. Percobaan pertama pada saat menggunakan optimizer Adam diperoleh nilai mAP sebesar 0,97073. Percobaan tersebut, dalam pengoptimalan pada dataset sehingga diperoleh nilai mAP sedemikian rupa yaitu dengan mempelajari bobot secara adaptif yang diperbarui dalam setiap epoch. Percobaan kedua dengan menggunakan optimizer SGD (*Stochastic Gradient Descent*) diperoleh nilai mAP sebesar 0,97355. Dalam optimizer SGD ini akan terus melakukan update dalam setiap satu data. Sehingga hasil yang diperoleh dalam mengklasifikasi kelas yang ada menjadi lebih akurat dan dapat dikenai dengan lebih cepat. Berdasarkan tabel 4.3 dari hasil pengujian optimizer dapat disimpulkan bahwa proses *training* yang dilakukan dengan menggunakan optimizer SGD lebih baik dari pada menggunakan optimizer Adam. Hal ini terbukti saat pada percobaan kedua pada tabel 4.3 pengujian dengan menggunakan optimizer SGD nilai mAP yang diperoleh sebesar 0,97355.

4.3.4 Pengujian Model YOLOV5

Pengujian model YOLOV5 juga dilakukan dalam proses *training*. Pengujian model YOLO dikombinasikan dengan beberapa hyperparameter seperti Epoch, Image, Batch, dan Optimizer. Model YOLOV5 yang digunakan dalam proses

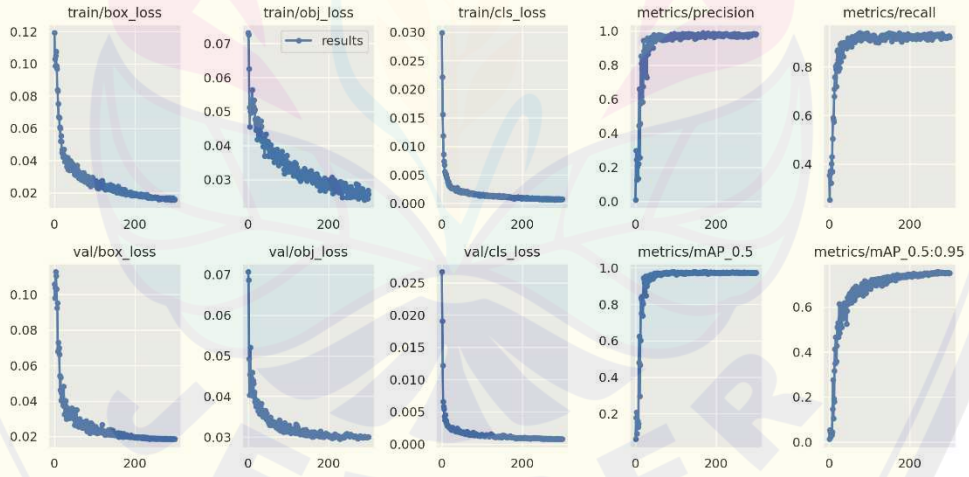
pengujian yaitu YOLOV5s, YOLOV5m dan YOLOV5x. Pengujian model ini bertujuan untuk membandingkan hasil terbaik dari ketiga model serta parameter yang digunakan. Pengujian terbaik yang dihasilkan kemudian akan dipilih dan bobot tersebut akan digunakan pada pengujian *real time*.

Tabel 4.4 Hasil pengujian model YOLOV5

NO	IMG	EPOCH	BATCH	MODEL	OPTIMIZER	mAP
1	640	300	8	YOLOV5s	SGD	0,97355
2	640	300	8	YOLOV5m	SGD	0,97699
3	640	300	8	YOLOV5x	SGD	0,98214

a. Pengujian model YOLOV5s

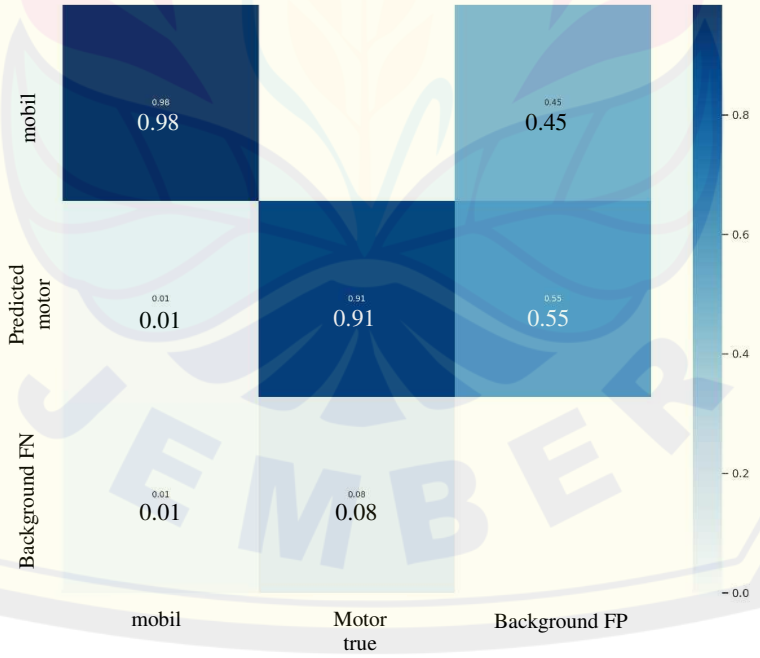
Berdasarkan pengujian dari model YOLOV5s diperoleh hasil *mean average Precision* (mAP) sebesar 0.97. Berikut merupakan grafik yang dihasilkan dari pengujian model YOLOV5s dengan menggunakan parameter image size 640, Epoch 300, Batch 8, serta Optimizer SGD dapat dilihat pada gambar 4.6 dibawah ini:



Gambar 4.6 Hasil *training* menggunakan model YOLOV5s

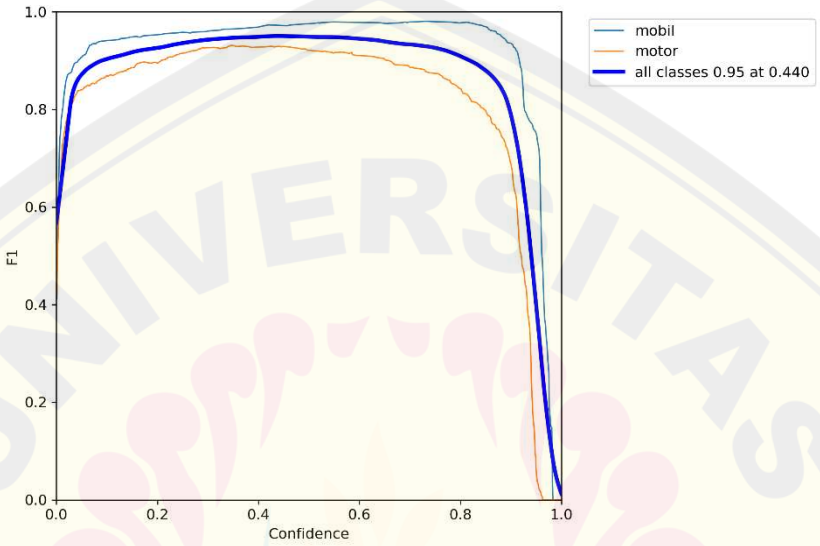
Berdasarkan gambar 4.6 diatas dihasilkan beberapa grafik *training* dan validasi diantaranya grafik bounding box, objek, kelas, mAP, *Precision* dan Recall. Berdasarkan grafik box_loss, objek_loss dan class_loss mengalami penurunan secara eksponensial, hal ini dipengaruhi oleh banyaknya Epoch pada saat

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 300, 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 0.5, mAP 0.5:0.95, *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 ketiga grafik tersebut, maka dapat dikatakan sistem yang dibuat semakin baik. Adapun nilai yang diperoleh dari grafik mAP, *Precision* dan Recall yaitu untuk mAP 0.5 sebesar 0.97, mAP 0.5:0.95 sebesar 0.75, nilai *Precision* diperoleh sebesar 0.98 dan Recall sebesar 0.92.



Gambar 4.7 Confusion matrix

Gambar 4.7 merupakan hasil dari Confusion matrix yang dihasilkan dari proses training dan validasi pada saat pengujian model YOLOV5s. Confusion matrix di atas menunjukkan bahwa YOLOV5s dapat memprediksi suatu kelas diantaranya mobil dengan akurasi rata-rata sebesar 98% dan memprediksi kelas motor dengan akurasi rata-rata sebesar 91%.

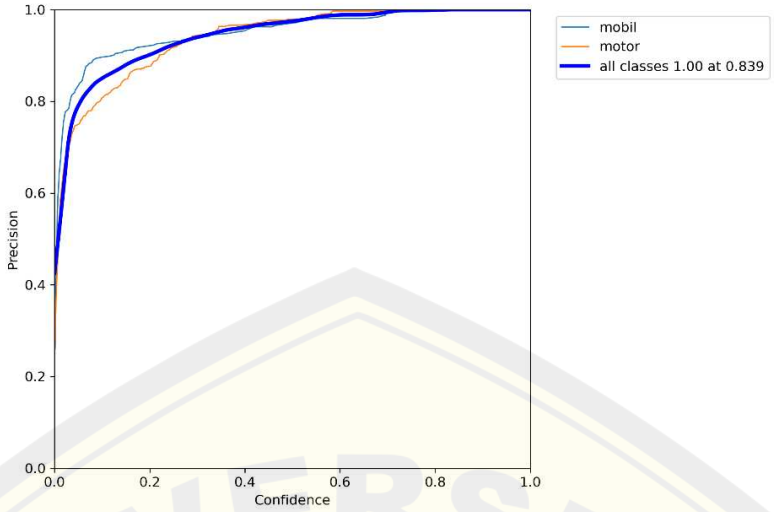


Gambar 4.8 Kurva nilai F1 terhadap nilai *Confidence*

Gambar 4.8 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 Presisi dan Recall. Adapun rumus dari F1 Score sendiri dapat dilihat seperti pada persamaan dibawah ini:

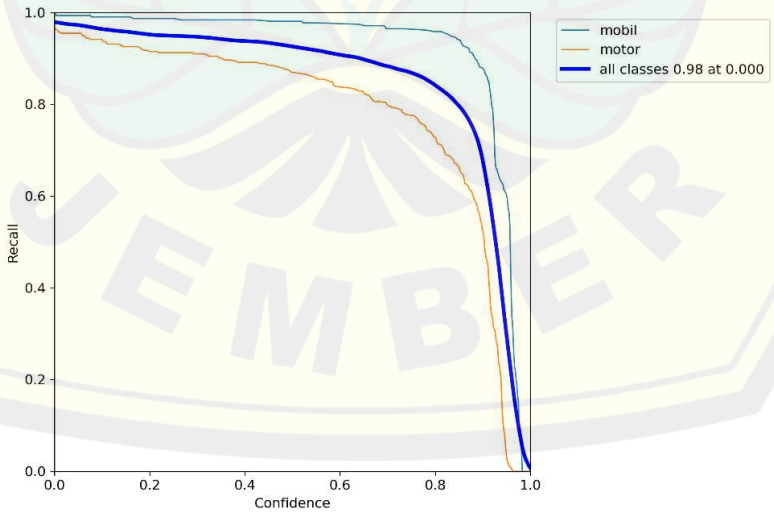
$$F1\ Score = \frac{2\ x\ (Precision\ x\ Recall)}{Precision+Recall} \quad (4.4)$$

Nilai F1 Score mencapai titik tertinggi sebesar 0.95 setiap kelasnya pada saat nilai *Confidence* 0.440.



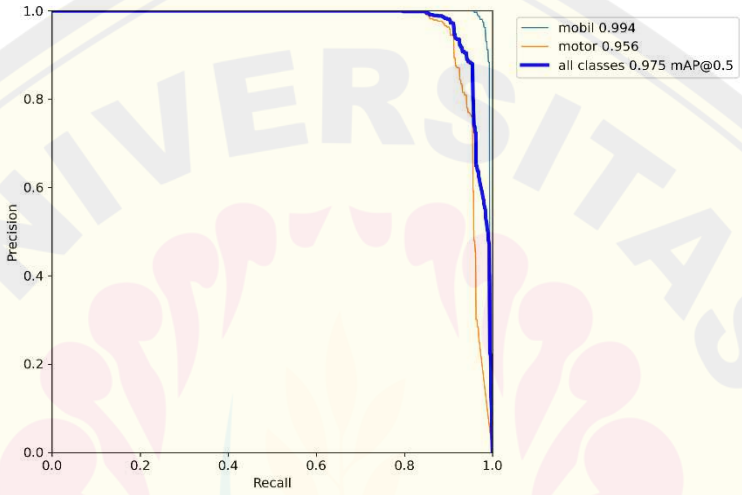
Gambar 4.9 Kurva nilai *Precision* terhadap nilai *Confidence*

Gambar 4.9 menunjukkan grafik antara nilai *Confidence* terhadap *Precision*. Dapat dilihat pada grafik yang dihasilkan dimana kelas motor dan mobil mencapai nilai *Precision* tertinggi pada saat *Confidence* skornya berada pada nilai 0.839. berdasarkan dari grafik yang dihasilkan dapat dikatakan bahwa semakin cepat nilai kelas motor dan kelas mobil 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 4.10 Kurva nilai Recall terhadap nilai *Confidence*

Gambar 4.10 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 0.98 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.

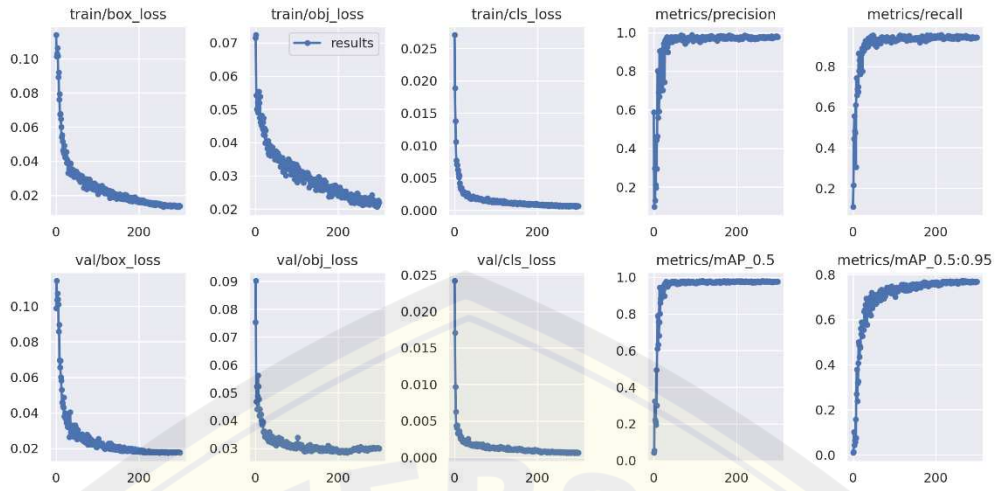


Gambar 4.11 Kurva nilai *Precision* terhadap nilai Recall

Gambar 4.11 menunjukkan grafik antara nilai *Precision* dan Nilai Recall. Berdasarkan dari grafik yang dihasilkan dapat dilihat bahwa nilai tertinggi dari kelas motor dan mobil masing-masing 0.956 dan 0.994 pada saat nilai mAP 0.5.

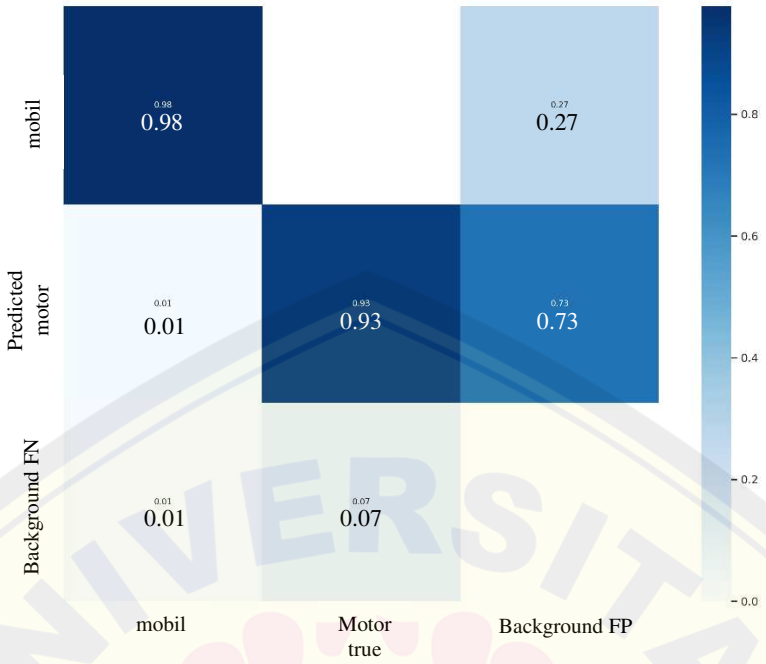
b. Pengujian model YOLOV5m

Pengujian dari model YOLOV5m didapatkan hasil *mean average Precision* (mAP) sebesar 0.976. Berdasarkan pengujian dari model YOLOV5m dengan menggunakan parameter image size 640, Epoch 300, Batch 8, dan Optimizer SGD diperoleh grafik seperti pada gambar 4.12 dibawah ini:



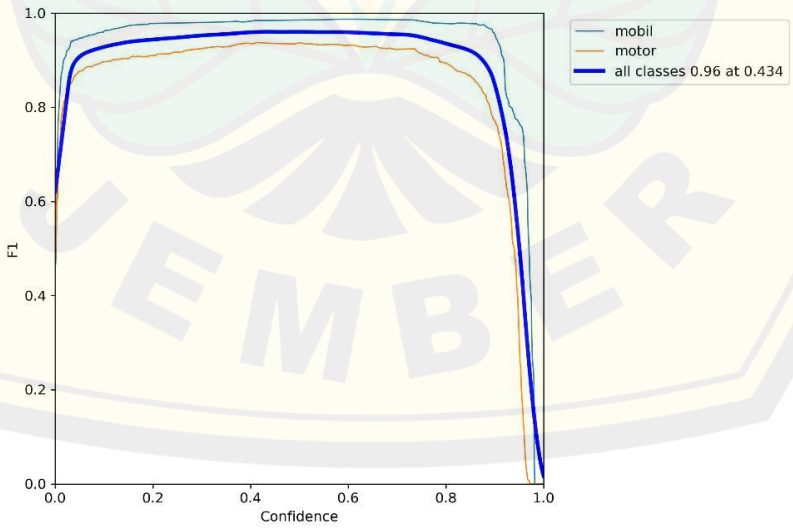
Gambar 4.12 Hasil *training* menggunakan model YOLOV5m

Gambar 4.12 diatas dihasilkan grafik-grafik *training* dan validasi diantaranya grafik bounding box, objek, kelas, mAP, *Precision* dan Recall. 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*. Epoch yang dipilih pada *training* dan validasi saat menggunakan model YOLOV5m yaitu sebanyak 300 Epoch, hal ini karena dianggap nilai tersebut merupakan nilai yang tidak terlalu memberatkan sistem serta nilai loss yang dihasilkan tidak terlalu besar. Grafik mAP 0.5, mAP 0.5:0.95, *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. Adapun nilai yang diperoleh dari grafik mAP, *Precision* dan Recall yaitu untuk mAP 0.5 sebesar 0.976, mAP 0.5:0.95 sebesar 0.768, nilai *Precision* diperoleh sebesar 0.97 dan Recall sebesar 0.94.



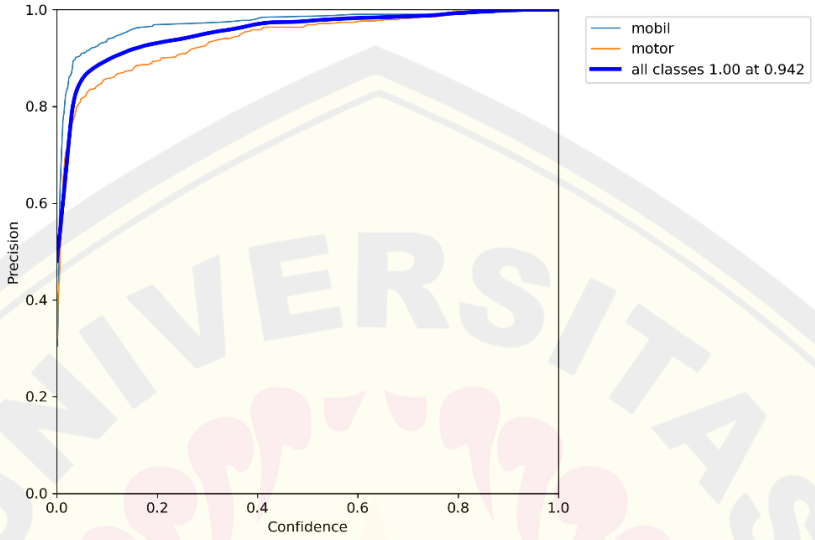
Gambar 4.13 Confusion matrix

Gambar 4.13 menunjukkan hasil dari Confusion matrix yang diperoleh dari hasil *training* dan validasi dari sistem dengan memakai salah satu model dari YOLOV5 yaitu YOLOV5m. Berdasarkan confusion matrix yang diperoleh dapat dilihat bahwa sistem dapat memprediksi kelas mobil dengan rata-rata akurasi sebesar 98% dan untuk kelas motor, sistem dapat memprediksinya dengan rata-rata akurasi sebesar 93%.



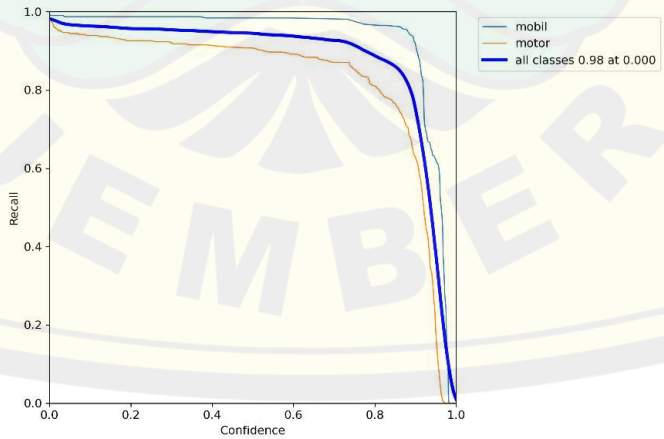
Gambar 4.14 Kurva nilai F1 terhadap nilai *Confidence*

Gambar 4.14 merupakan grafik yang dihasilkan dari pengujian model kedua yaitu YOLOV5m dimana nilai F1 skor mencapai nilai tertinggi pada masing-masing kelasnya dengan nilai 0.96 pada saat *Confidence* skor berada pada nilai 0.434.



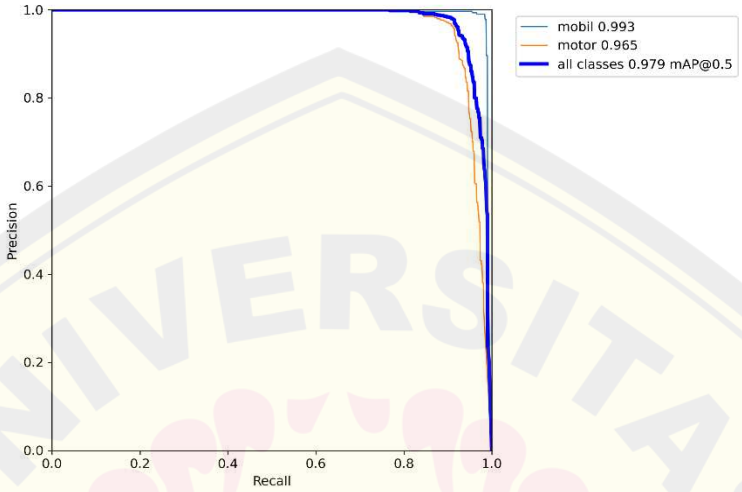
Gambar 4.15 Kurva nilai *Precision* terhadap nilai *Confidence*

Gambar 4.15 menunjukkan grafik antara nilai *Precision* dan *Confidence*. Berdasarkan dari grafik tersebut nilai *Precision* dari setiap kelas mencapai titik tertinggi yaitu pada saat nilai *Confidence* berada pada nilai 0.942. semakin besar nilai *Confidence*, maka akan akan semakin tinggi nilai *Precision* yang dihasilkan begitu juga sebaliknya.



Gambar 4.16 Kurva nilai Recall terhadap nilai *Confidence*

Gambar 4.16 merupakan grafik antara nilai *Confidence* dengan nilai Recall. Dapat dilihat pada gambar grafik tersebut merupakan kebalikan dari grafik *Precision* sebelumnya, dimana nilai Recall tertinggi dari kelas motor dan mobil 0.98.

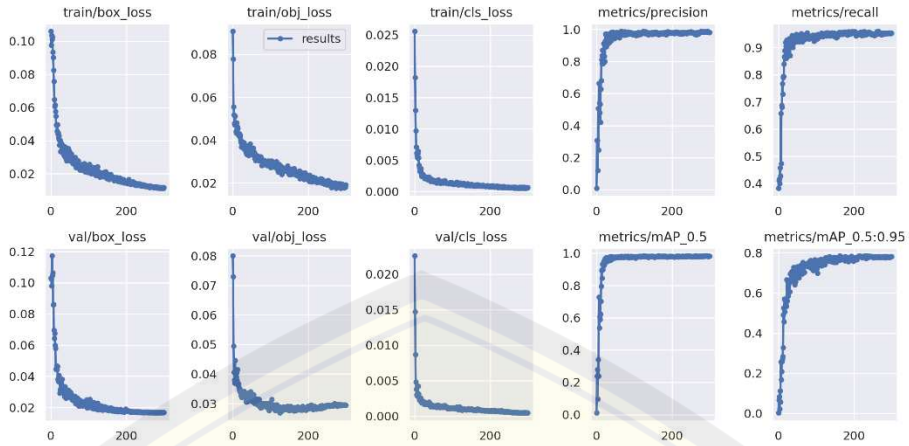


Gambar 4.17 Kurva nilai *Precision* terhadap nilai Recall

Gambar 4.17 merupakan grafik hubungan antara nilai *Precision* dan nilai Recall yang dihasilkan saat proses *training* dan validasi. Berdasarkan grafik tersebut nilai pada setiap kelasnya mencapai 0.979 pada saat mAP0.5. Recall ini memiliki arti yaitu antara nilai *True Positive* dan nilai prediksi yang positif untuk model prediksi dengan menggunakan ambang probabilitas yang berbeda.

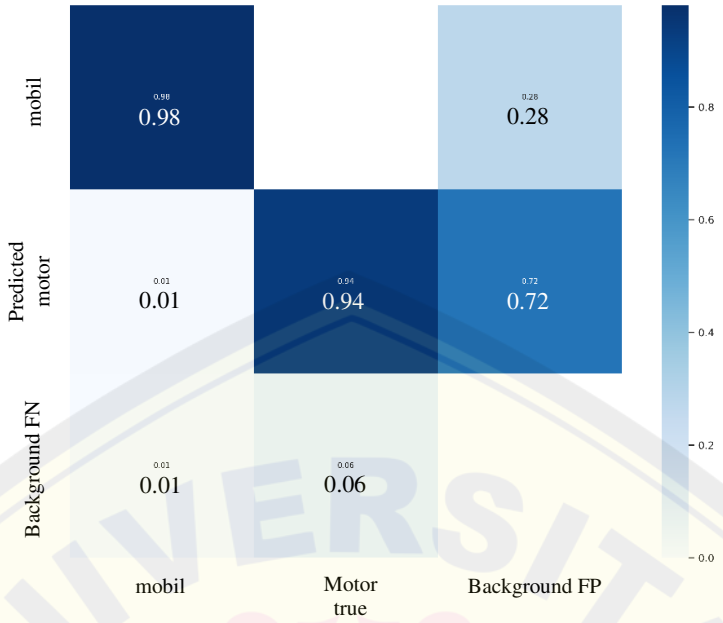
c. Pengujian model YOLOV5x

Terakhir, untuk pengujian dari model yaitu menggunakan model YOLOV5x. Penggunaan model YOLOV5x diperoleh hasil *mean average Precision* (mAP) sebesar 0.98. Adapun grafik yang dihasilkan dari pengujian model YOLOV5x dengan menggunakan hyperparameter yang sama seperti pada pengujian model sebelumnya yaitu dengan image size 640, Epoch 300, Batch 8 dan Optimizer SGD dapat dilihat pada gambar 4.18 dibawah ini:



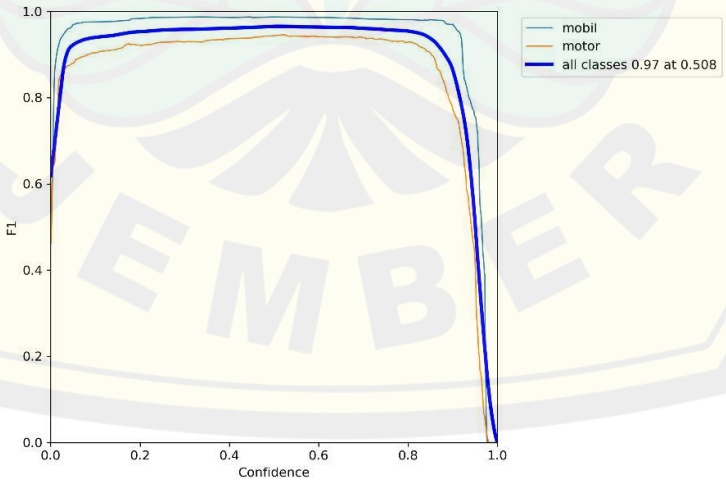
Gambar 4.18 Hasil *training* menggunakan model YOLOV5x

Gambar 4.18 diatas dihasilkan beberapa grafik *training* dan validasi diantaranya grafik bounding box, objek, kelas, mAP, *Precision* dan Recall. Sama seperti pengujian model yang sebelumnya, grafik box_loss, objek_loss dan class_loss mengalami penurunan secara eksponensial dan untuk grafik mAP 0.5, mAP 0.5:0.95, *Precision* dan Recall kebalikan dari ketiga grafik sebelumnya. Epoch yang digunakan pada pengujian model ini yaitu menggunakan variasi Epoch 100 dan Epoch 300. Namun, penggunaan Epoch 100 masih diperoleh hasil yang kurang optimal. Oleh sebab itu, dipilihlah model dengan menggunakan Epoch sebesar 300. Hal ini karena penggunaan Epoch 300 diperoleh nilai-nilai yang cukup akurat dan tidak terlalu memberatkan sistem. 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 ketiga grafik tersebut, maka dapat dikatakan sistem yang dibuat semakin baik. Adapun hasil nilai yang diperoleh dari grafik mAP, *Precision* dan Recall yaitu untuk mAP 0.5 sebesar 0.98, mAP 0.5:0.95 sebesar 0.78, nilai *Precision* diperoleh sebesar 0.98 dan Recall sebesar 0.95.



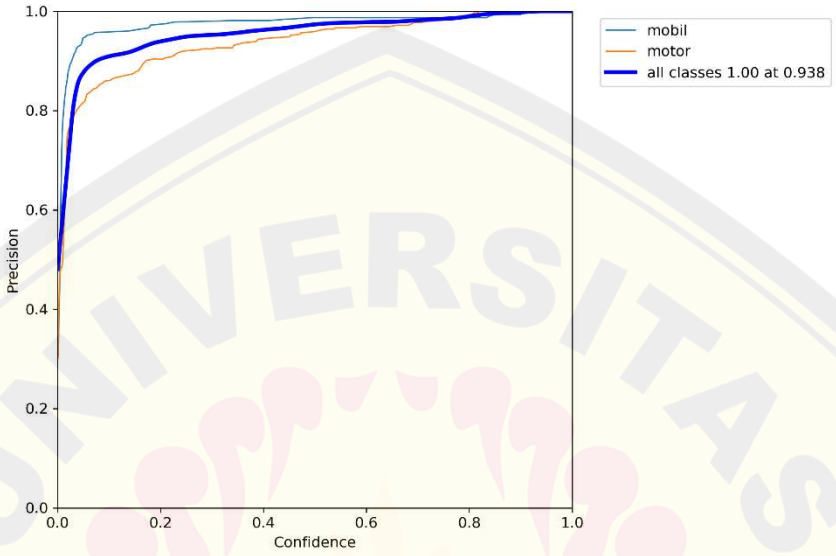
Gambar 4.19 Confusion matrix

Gambar 4.19 merupakan hasil Confusion matrix yang diperoleh ketika melakukan proses *training* dan validasi dari sistem pada saat menggunakan model YOLOV5x. Model yang digunakan dalam proses pelatihan juga dikombinasikan dengan beberapa hyperparameter seperti Epoch, Batch dan sebagainya. Hasil yang diperoleh sistem saat menggunakan model YOLOV5x dengan prediksi mobil dihasilkan nilai rata-rata akurasi sebesar 98% dan nilai rata-rata akurasi prediksi untuk kelas motor dihasilkan sebesar 94%.



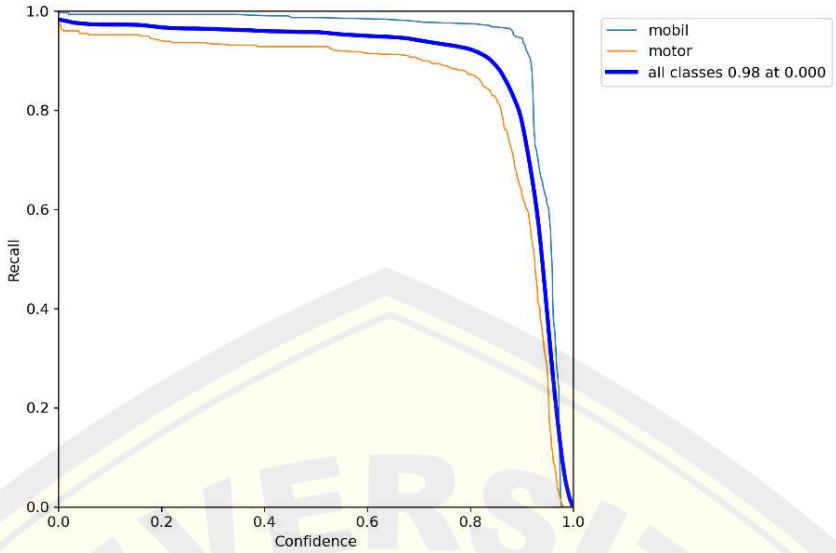
Gambar 4.20 Kurva nilai F1 terhadap nilai *Confidence*

Gambar 4.20 menunjukkan grafik antara nilai F1 Score terhadap nilai *Confidence* yang dihasilkan setelah melalui proses *training* dan validasi. Dapat dilihat pada grafik tersebut, nilai tertinggi dari kelas motor dan mobil berada pada saat *Confidence* skornya 0.508.



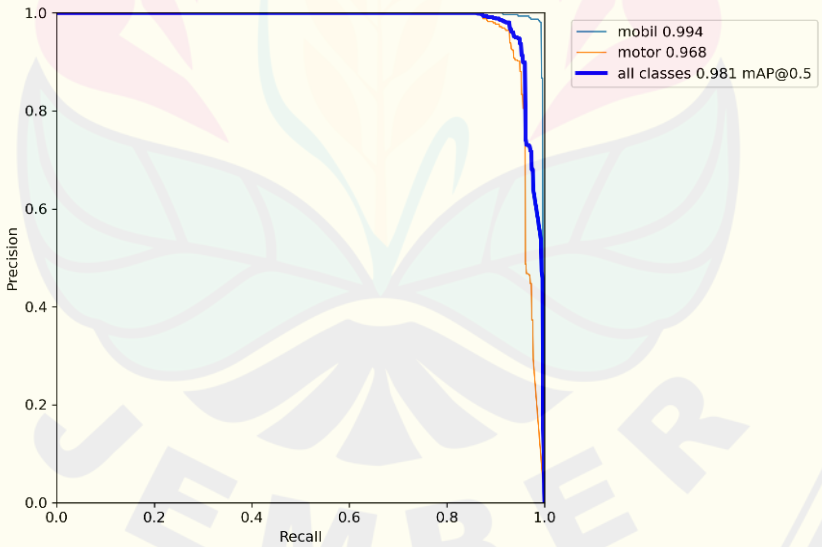
Gambar 4.21 Kurva nilai *Precision* terhadap nilai *Confidence*

Gambar 4.21 menunjukkan nilai *Confidence* terhadap nilai *Precision* yang dihasilkan. Pengujian model YOLOV5 terakhir ini nilai *Precision* kelas mencapai 1.0 pada saat nilai *Confidence* 0.938. Matrix *Precision* ini digunakan apabila kondisi *False Positive* mempengaruhi sistem, misalnya pesan penting namun sistem memprediksi spam.



Gambar 4.22 Kurva nilai Recall terhadap nilai *Confidence*

Gambar 4.22 merupakan hasil grafik yang diperoleh dari proses *training* dan validasi antara nilai *Confidence* dengan nilai Recall. Dapat dilihat pada gambar grafik tersebut, dimana nilai Recall tertinggi dari kelas motor dan mobil 0.98.



Gambar 4.23 Kurva nilai *Precision* terhadap nilai Recall

Gambar 4.23 merupakan grafik hubungan antara nilai *Precision* dan nilai Recall yang dihasilkan saat proses *training* dan validasi. Berdasarkan grafik tersebut nilai pada setiap kelasnya mencapai 0.981 pada saat mAP0.5. Recall ini

memiliki arti yaitu antara nilai *True Positive* dan nilai prediksi yang positif untuk model prediksi dengan menggunakan ambang probabilitas yang berbeda.

Pengujian model yang telah dilakukan dengan 3 varian model yang digunakan dalam proses *training* dan validasi sistem dihasilkan prediksi untuk nilai rata-rata akurasi dari kelas mobil sebesar 98% dari ketiga model yang digunakan. Sedangkan prediksi kelas motor dihasilkan nilai rata-rata akurasi sebesar 91% pada saat menggunakan model YOLOV5s, 93% saat menggunakan model YOLOV5m, dan 94% saat menggunakan model YOLOV5x. Berdasarkan hasil-hasil yang telah dijabarkan diatas, dapat disimpulkan bahwa YOLOV5x merupakan model yang menghasilkan nilai presisi terbaik dari percobaan yang telah dilakukan. Namun berdasarkan hasil tersebut belum tentu model YOLOV5x signifikan. Oleh karena itu perlu dilakukan pengujian signifikansi dari semua model yang digunakan. Pengujian signifikansi ini dilakukan dengan ANOVA (*Analysis of Variance*). Berikut merupakan hasil yang diperoleh pada saat pengujian ANOVA.

Tabel 4.5 Sebaran data YOLOV5

	Model		
	YOLOV5x	YOLOV5s	YOLOV5m
SGD, 16, 100	0,78	0,736	0,76
	0,98	0,97	0,97
	0,969	0,969	0,96
	0,95	0,91	0,94
SGD, 8, 300	0,78	0,75	0,768
	0,98	0,97	0,976
	0,98	0,98	0,97
	0,95	0,92	0,94
Adam, 16, 300	0,747	0,748	0,756
	0,97	0,97	0,97
	0,968	0,958	0,94
	0,93	0,92	0,94

Adam, 8, 100	0,717	0,719	0,706
	0,967	0,959	0,96
	0,95	0,926	0,97
	0,92	0,927	0,89

Berdasarkan Tabel 4.5 diatas merupakan hasil yang diperoleh ketika melakukan proses *training* sebanyak 12 kali percobaan dengan menggunakan hyperparameter yang berbeda-beda dalam setiap proses *training*nya. Adapun parameter yang digunakan dalam proses *training* diantaranya yaitu Epoch, Batch, Optimizer dan Model YOLOV5. Penggunaan image size menggunakan ketetapan sebesar 640. Hasil yang diperoleh dalam proses *training* ini yaitu nilai mAP 0.5; mAP 0,5:0,95; *Precision* dan Recall.

Tabel 4.6 Hasil Pengujian ANOVA

Source of Variation	SS	df	MS	F	P-Value	F crit
Sample	0,005824	3	0,001941	0,175011	0,912611	2,866266
Columns	0,001341	2	0,000671	0,060449	0,941437	3,259446
Interaction	0,000533	6	8,89E-05	0,008013	0,999997	2,363751
Within	0,399363	36	0,011093			
Total	0,407062	47				

Berdasarkan pengujian ANOVA dengan melihat nilai P-Value, F dan F crit dimana nilai P-Value dan nilai F lebih besar dari nilai toleransi kesalahan (0,05), dapat dikatakan bahwa perbedaan variabel yang digunakan tidak mempengaruhi output yang dihasilkan, serta ditunjang dengan referensi-referensi yang ada, maka dipilihlah model YOLOV5s. Hal ini karena untuk hasil yang diperoleh dari model YOLOV5s selisihnya tidak jauh berbeda dengan hasil yang diperoleh dari model YOLOV5m dan YOLOV5x. Selain itu untuk bobot yang dihasilkan dari YOLOV5s

jauh lebih ringan daripada bobot yang dihasilkan saat penggunaan model YOLOV5m dan YOLOV5x.

4.4 Pengujian Kamera

Terdapat dua tahapan yang dilakukan dalam proses pengujian kamera yang mana meliputi resolusi video dan pencahayaan. Resolusi video merupakan ukuran dari video yang digunakan dalam pembuatan sistem. Sedangkan pengujian pencahayaan yang dilakukan bertujuan untuk mengetahui pengaruh intensitas cahaya dari sistem dalam mendeteksi suatu objek dan mengklasifikasikannya berdasarkan kelas yang ada.

a. Resolusi Video

Pengujian resolusi video dan kamera disesuaikan dengan video CCTV yang telah diberikan oleh dinas perhubungan kabupaten Jember. Pengujian resolusi ini akan berpengaruh terhadap besar kecilnya ketajaman dari suatu gambar yang telah dihasilkan. Selain berpengaruh terhadap ketajaman dari suatu gambar resolusi video ini juga berpengaruh terhadap jumlah bit data dan proses transfer datanya. Sehingga besar kemungkinan akan berpengaruh dari perubahan informasi yang terkandung di dalamnya. Video kamera yang digunakan dalam pengujian sistem ini menggunakan ukuran 1920 *pixel* x 1080 *pixel*.

Gambar 4.24 Video CCTV dengan ukuran 1280 pixel x 720 pixel

Gambar 4.24 diatas menunjukkan hasil dari rekaman video CCTV dinas perhubungan kabupaten Jember dengan ukuran 1280 pixel x 720 pixel. Penelitian yang dilakukan ini yaitu membuat sistem yang dapat mendeteksi objek berupa kendaraan dengan membedakan berdasarkan kelas yang ada, serta memprediksi kecepatan kendaraannya dengan menggunakan rumus matematis yang telah disiapkan sebelumnya. Jadi dengan ukuran resolusi yang sedemikian rupa sudah cocok digunakan untuk pembuatan sistem dalam penelitian ini.

b. Pencahayaan

Pengujian pencahayaan ini dilakukan dengan mengukur intensitas cahaya pada waktu yang telah ditentukan berdasarkan dataset yang digunakan. Pengukuran intensitas cahaya ini menggunakan alat ukur lux meter. Pengujian pencahayaan ini sangat penting dilakukan, karena besarnya intensitas cahaya akan mempengaruhi akurasi sistem dalam mendeteksi suatu objek. Oleh karena ini penelitian yang dilakukan mengukur intensitas cahaya pada pagi, siang, sore dan malam hari. Adapun pengujian pencahayaan yang telah dilakukan dapat dilihat pada tabel 4.7 Pengaruh intensitas cahaya terhadap tampilan gambar.

Tabel 4.7 Pengaruh intensitas cahaya terhadap tampilan gambar

No.	Waktu (WIB)	Intensitas Cahaya (lux)	Tampilan Gambar
1.	08.01	7170	

Gambar 4.25 Hasil CCTV pada intensitas cahaya 7170

2.	12.43	15810	 Gambar 4.26 Hasil CCTV pada intensitas cahaya 15810
3.	15.27	11240	 Gambar 4.27 Hasil CCTV pada intensitas cahaya 11240
4.	19.00	205	 Gambar 4.28 Hasil CCTV pada intensitas cahaya 205

Berdasarkan tabel 4.7 merupakan hasil dari pengujian intensitas cahaya pada lokasi area yang terpantau CCTV. pengujian tersebut dilakukan sebanyak 4 kali pengujian dengan waktu yang berbeda-beda sesuai dengan apa yang telah direncanakan pada metode penelitian. Adapun pengujian ini menggunakan alat ukur lux meter untuk mendeteksi suatu besaran dari intensitas cahaya. Pengujian pertama, dilakukan pada pagi hari pada waktu 08.01 diperoleh intensitas cahaya sebesar 7170 lux. Pengujian kedua, dilakukan pada siang hari pada waktu 12.43 diperoleh nilai intensitas cahaya sebesar 15810 lux. Pengujian ketiga, dilakukan pada sore hari pada waktu 15.25 diperoleh nilai intensitas cahaya sebesar 11240 lux. Pengujian terakhir, dilakukan pada malam hari pada waktu 19.00 diperoleh

intensitas cahaya sebesar 205 lux. Berdasarkan nilai intensitas cahaya yang telah didapatkan selama pengujian dapat disimpulkan bahwa dalam keperluan sistem untuk mendeteksi suatu objek yang cocok dengan intensitas cahaya yaitu pada pagi, siang dan sore hari. Hal ini karena nilai intensitas yang diperoleh cukup besar. Sedangkan pada malam hari nilai intensitas cahaya yang terbaca sangat kecil, hal ini akan berpengaruh terhadap sistem dalam mendeteksi objek, bisa jadi objek tersebut salah pendeteksian atau bahkan tidak terdeteksi.

4.5 Pengujian Pendeteksian Kendaraan

Pengujian pendeteksian kendaraan ini dilakukan dengan memanfaat video rekaman CCTV yang telah diberikan oleh dinas perhubungan Kabupaten Jember. Pengujian ini akan mendeteksi objek berdasarkan kelas yang telah ditentukan. Kelas kendaraan yang akan dideteksi dalam penelitian ini hanya dibatasi sebanyak dua kelas, yaitu mobil dan motor. Pengujian deteksi kendaraan ini dilakukan dengan menggunakan beberapa video dengan durasi ± 2 menit yang berbeda-beda berdasarkan intensitas cahaya yang diperoleh. Adapun bobot model yang digunakan pada pengujian pendeteksian kendaraan ini yaitu bobot yang dihasilkan pada proses *training* dengan *Image Size* 640, *Epoch* 300, *Batch size* 8, dengan model yang dipakai YOLOV5s dan *Optimizer* SGD. Berikut merupakan tabel dari hasil pengujian deteksi kendaraan dapat dilihat pada tabel 4.8 Hasil pengujian pendeteksian kendaraan dibawah ini:

Tabel 4.8 Hasil pengujian pendeteksian kendaraan

Pengujian ke-	Hitung Manual		Hasil Deteksi		Error Persen Mobil(%)	Error Persen Motor(%)
	Motor	Mobil	Motor	Mobil		
1	41	4	39	4	0	4.87
2	60	3	57	3	0	5
3	8	4	7	4	0	12.5
4	75	4	62	4	0	17.3
Rata-rata					0%	9.91%
Standar Deviasi					0	6.08

Tabel 4.8 menampilkan hasil pengujian dari deteksi kendaraan yang dibedakan berdasarkan kelas yang telah ditentukan sebelumnya. Tabel tersebut memperlihatkan pengujian dari perbandingan antara pembacaan deteksi oleh sistem dengan perhitungan manual dari kelas mobil dan motor yang melintas. Adapun pengujian ini dilakukan dengan menggunakan beberapa video yang berbeda-beda. Pengujian pertama untuk pada perhitungan kelas mobil dan motor secara manual diperoleh sebanyak 41 untuk kelas motor dan 4 untuk kelas mobil, pengujian kedua untuk kelas motor diperoleh sebanyak 60 dan kelas mobil sebanyak 3, selanjutnya pada pengujian ketiga diperoleh masing-masing untuk kelas motor dan kelas mobil sebanyak 8 dan 4, terakhir pada pengujian keempat diperoleh sebanyak 75 motor dan 4 mobil.

Perolehan hitung manual yang telah didapat akan dibandingkan dengan hasil dari pembacaan sistemnya, sehingga dihasilkan nilai error persen dari kedua kelas tersebut. Dari keempat pengujian yang telah dilakukan untuk kelas mobil antara hitung manual dan pembacaan sistemnya sama, sehingga untuk error persen dari mobil dari keempat pengujian tersebut diperoleh sebesar 0%. Pengujian pertama untuk kelas motor berdasarkan hasil deteksi dari sistem diperoleh sebanyak 39 motor, sehingga diperoleh nilai error persen dari kelas motor pada pengujian pertama sebesar 4,87%, pengujian kedua sistem mendeteksi kelas motor sebanyak 57 selisih 3 motor dari perhitungan manualnya, sehingga didapat error persen sebesar 5%. Pengujian ketiga untuk kelas motor hanya selisih 1 motor dari pembacaan sistemnya, namun diperoleh error persen yang cukup besar yaitu 12,5%. Hal ini karena pada pengujian ketiga untuk video yang digunakan berdurasi pendek. Jadi antara perhitungan manual dan sistem hanya diperoleh sedikit. Pengujian yang terakhir diperoleh nilai error persen yang cukup besar yaitu 17,3% pada kelas motor. Hal ini dikarenakan selisih yang dihasilkan cukup besar antara hasil deteksi sistem dengan perhitungan manualnya yaitu sebesar 13 motor. Selisih yang dihasilkan dari pengujian ini dipengaruhi oleh berbagai faktor diantaranya adanya sebuah mobil atau motor yang saling berdekatan sehingga dari dua atau lebih motor atau mobil yang berdekatan tersebut terbaca oleh sistem hanya 1 buah kendaraan saja. Selain pengaruh dari berdekatannya suatu kendaraan, faktor lainnya yang

sangat mempengaruhi dari kesalahan sistem dalam mendeteksi suatu objek yaitu intensitas cahaya. Besarnya intensitas cahaya sangat mempengaruhi sistem dalam mendeteksi suatu objek oleh karena itu perlu dilakukan pengujian dengan besar intensitas cahaya yang bervariasi sesuai dengan waktu yang telah ditentukan.

Tabel 4.9 Hasil pengujian pengaruh intensitas cahaya terhadap deteksi kendaraan

Pengujian ke-	Intensitas Cahaya (lux)	Hitung Manual		Hasil Deteksi		Error Persen Mobil(%)	Error Persen Motor(%)
		Motor	Mobil	Motor	Mobil		
1	7170	55	18	52	18	0	5.45
2	15810	63	22	63	22	0	0
3	11240	87	24	83	24	0	4.59
4	205	44	23	34	7	69.56	22.72
Rata-rata						17.39%	8.19%
Standar Deviasi						34.78	9.98

Hasil pengujian pengaruh intensitas cahaya terhadap deteksi kendaraan yang telah didapat pada tabel 4.9 diatas dapat disimpulkan bahwa sistem dapat mendeteksi kendaraan dengan baik pada rentang nilai intensitas cahaya antara 7000 lux hingga 15000 lux. Intensitas cahaya pada nilai 205 lux yang diambil pada malam hari dihasilkan deteksi kendaraan dari sistem yang tidak akurat. Sistem yang telah dibuat sulit mendeteksi suatu objek dengan mengelompokkan berdasarkan kelasnya pada nilai intensitas cahaya kurang dari 7000 lux.

4.6 Pengujian Kecepatan Kendaraan

Pengujian selanjutnya setelah pengujian deteksi kendaraan yaitu pengujian prediksi kecepatan kendaraan. Pengujian ini dilakukan dengan menggunakan video yang telah disediakan yang berdurasi sekitar 2 menit yang berbeda-beda intensitas cahaya sesuai dengan pengambilan waktu videonya. Adapun perhitungan kecepatan kendaraan yang digunakan pada sistem yang telah dibuat yaitu seperti persamaan yang ada di bawah ini:

$$v = \frac{s}{t} \quad (4.5)$$

v = kecepatan

s = jarak asli antara kedua garis awal dan garis akhir

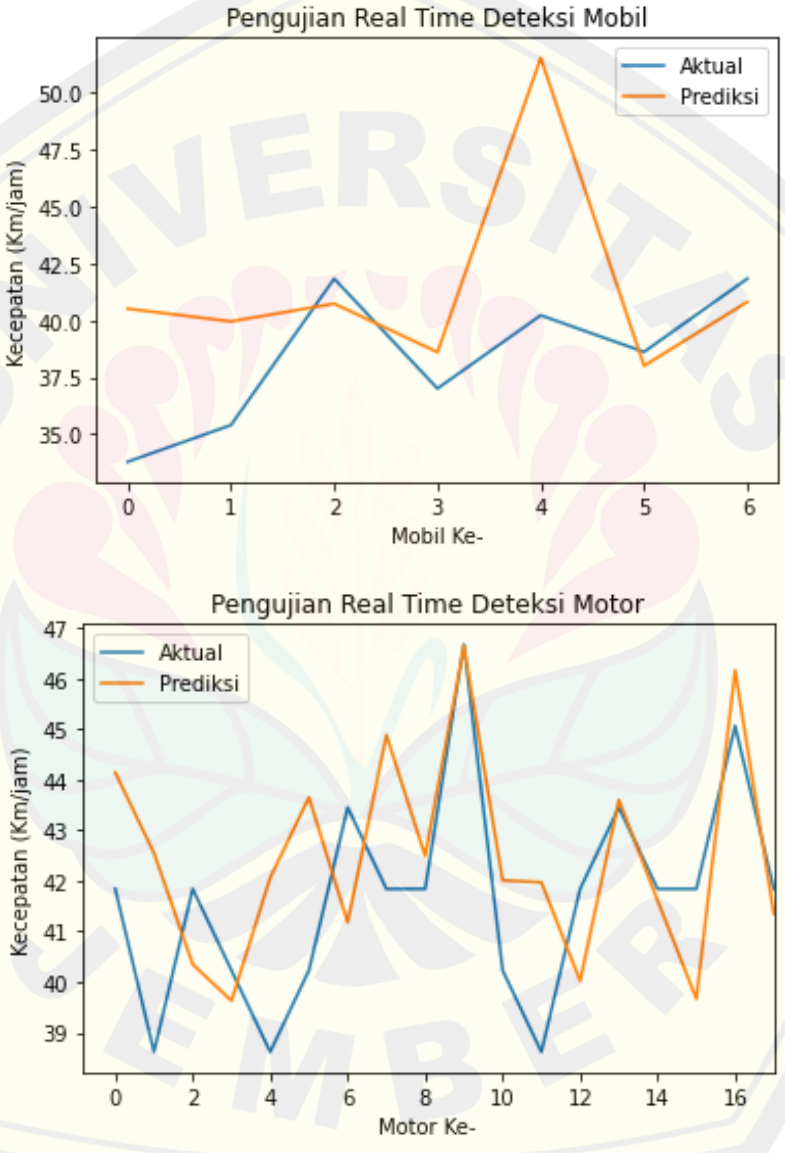
t = waktu yang dibutuhkan suatu objek untuk mencapai kedua garis tersebut

waktu yang dibutuhkan suatu objek untuk melewati garis diperoleh dari jumlah frame yang terdeteksi oleh sistem ketika objek melewati kedua garis. Kedua garis tersebut didapat dengan perhitungan secara langsung dari jarak asli yang digunakan. Kedua garis tersebut juga digunakan untuk memperoleh waktu dengan memanfaatkan nilai frame yang terdeteksi. Hasil dari pengujian prediksi kecepatan dapat dilihat pada Tabel 4.10 Hasil pengujian kecepatan kendaraan dibawah ini:

Tabel 4.10 Hasil pengujian kecepatan kendaraan

Pengujian ke-	Jenis Kendaraan	Perhitungan Manual Kecepatan (Km/Jam)	Hasil Deteksi Kecepatan (Km/Jam)	Error Persen (%)
1	Mobil	33,79	40,52	19,91
	Mobil	35,40	39,96	12,88
	Motor	41,84	44,14	5,49
	Motor	38,62	42,55	10,17
	Mobil	41,84	40,74	2,62
2	Mobil	37,01	38,61	4,32
	Motor	41,84	40,35	3,56
	Motor	40,23	39,63	1,49
	Motor	38,62	42,06	8,9
	Motor	40,23	43,65	8,5
3	Mobil	40,23	51,55	28,13
	Motor	43,45	41,18	5,22
	Motor	41,84	44,88	7,26
	Motor	41,84	42,49	2,40
	Motor	46,67	46,64	0,06
4	Mobil	38,62	38,01	1,57
	Motor	40,23	42,01	4,42
	Motor	38,62	41,97	8,67
	Motor	41,84	40,02	4,34
	Motor	43,45	43,6	0,34
5	Motor	41,84	41,61	0,54
	Motor	41,84	39,67	5,18

	Motor	45,06	46,16	2,44
	Mobil	41,84	40,82	2,43
	Motor	41,84	41,35	1,17
Rata-rata Error persen motor				4,45
Rata-rata Error persen mobil				10,26
Standar Deviasi Error Motor				3.23
Standar Deviasi Error Mobil				10.41



Gambar 4.29 Grafik pengujian kecepatan kendaraan

4.7 Pengujian *Running Text*

Pengujian *running text* dilakukan dengan tujuan agar menampilkan kecepatan kendaraan yang terdeteksi oleh sistem yang telah dibuat. Adapun output yang digunakan yaitu Max7219 led matrix display yang dihubungkan dengan arduino kemudian dari arduino tersebut dikomunikasikan dengan Jetson Nano dengan menggunakan pin komunikasi serial. Berdasarkan pembacaan serial tersebut akan ditampilkan hasil prediksi kecepatan kendaraan dari sistem pada led matrix Max7219. Berikut merupakan hasil pengujian yang diperoleh dari pengujian *running text* ini dapat dilihat pada Tabel 4.11 Hasil pengujian *running text* dibawah ini:

Tabel 4.11 Hasil pengujian *running text*

Pengujian Ke-	Prediksi kecepatan	Tingkat Keberhasilan	Gambar
1	34,75 km/jam	Berhasil	 Gambar 4.30 Keluaran *Running Text* pada pengujian pertama
2	34,98 km/jam	Berhasil	 Gambar 4.31 Keluaran *Running Text* pada pengujian kedua

3	35 km/jam	Berhasil	 Gambar 4.32 Keluaran *Running Text* pada pengujian Ketiga
4	36,16 km/jam	Berhasil	 Gambar 4.33 Keluaran *Running Text* pada pengujian keempat
5	34,47 km/jam	Berhasil	 Gambar 4.34 keluaran *Running Text* pada pengujian kelima

Hasil pengujian *Running text* pada tabel 4.11 diatas merupakan pengujian dimana ketika sistem dijalankan saat membaca sebuah video maupun secara *real time*. Pengujian tersebut akan memberikan keluaran pada matrix MAX7219 berupa angka dari prediksi sistem dalam membaca kecepatan kendaraan yang melintas. Pembacaan pada matrix MAX7219 dikirimkan melalui pembacaan pin serial Arduino. Adapun pin yang digunakan pada Arduino diantaranya; pin VCC 5V yang digunakan sebagai sumber; pin GND; pin 11 yang dihubungkan pada DIN pada Led matrix; pin 3 dihubungkan pada CS Led matrix; dan pin 13 yang dihubungkan pada CLK Led matrix.

4.8 Pengujian *Real Time*

Pengujian terakhir pada penelitian ini yaitu pengujian *real time* yang mana pengujian ini dilakukan untuk mengetahui tingkat keakuratan sistem yang telah dibuat menggunakan algoritma YOLOV5 dalam mendeteksi sebuah objek berdasarkan kelas dari kendaraan yang telah ditetapkan, serta memprediksi kecepatan dari masing-masing kendaraan yang melintas. Adapun pengujian *real time* yang dilakukan ini menggunakan Jetson Nano yang digunakan sebagai pengganti komputer dalam memproses suatu gambar. *Running Text* yang digunakan untuk menampilkan hasil prediksi kecepatan oleh sistem dalam bentuk angka, serta kamera Webcam yang digunakan untuk pengujian secara langsung. Berikut pada tabel 4.12 merupakan hasil dari pengujian *real time* yang telah dilakukan.

Tabel 4.12 Hasil pengujian *real time*

No	Gambar	Jenis kendaraan	Hasil perhitungan manual (Km/Jam)	Hasil Prediksi kecepatan (Km/jam)	Error Persen (%)
1	 Gambar 4.35 Hasil pengujian *real time* objek pertama	Motor	46	45,11	1,93

2	 Gambar 4.36 Hasil pengujian *real time* objek kedua	Mobil	42	42,67	1,59

3	 Gambar 4.37 Hasil pengujian *real time* objek ketiga	Motor	48	45,63	4,93

4	 Gambar 4.38 Hasil pengujian *real time* objek keempat	Motor	37	38,98	5,35

5	 Gambar 4.39 Hasil pengujian *real time* objek kelima	Mobil	34	38,41	12,97

6	 Gambar 4.40 Hasil pengujian *real time* objek keenam	Motor	35	34,51	1,4

7	 Gambar 4.41 Hasil pengujian *real time* objek ketujuh	Motor	32	31,59	1,28

8	 Gambar 4.42 Hasil pengujian *real time* objek kedelapan	Motor	39	40,82	4,66

9	 Gambar 4.43 Hasil pengujian *real time* objek kesembilan	Mobil	42	37,32	11,14

10	 Gambar 4.44 Hasil pengujian *real time* objek kesepuluh	Motor	41	39,29	4,17

Hasil pengujian real time yang telah didapat pada tabel 4.12 merupakan hasil yang didapat dengan melakukan pengujian secara langsung di lokasi menggunakan laptop. Pengambilan data tersebut dilakukan pada siang hari dengan rentang intensitas cahaya yang telah dilakukan sebelumnya. Lokasi pengambilan data tersebut tepat di samping Monumen patung Triumviraat. Penulis memilih tempat tersebut karena lokasinya yang cukup baik dalam pengambilan data. Penentuan jarak dari kedua garis yang telah ditentukan juga dipengaruhi oleh tingginya penempatan kamera yang akan digunakan. Jika penempatan kamera sejajar dengan kendaraan, maka untuk kedua garis yang akan ditentukan dalam penentuan jarak tersebut akan memendek atau bahkan berimpit, hal ini akan mempengaruhi pembacaan frame berdasarkan objek yang melintas oleh sistem. Hasil terbaik yang diperoleh selama pengujian *real time* ini yaitu pada hasil pengujian *real time* objek ketujuh dengan error persen sebesar 1,28%. Hal ini karena pada pengujian tersebut jalan tersebut hanya 1 kendaraan yang melintas (dalam keadaan sepi). Sedangkan hasil error persen terbesar didapatkan pada pengujian *real time* kelima dan kesembilan yaitu sebesar 12,97% dan 11,14%. Hal ini karena saat pengujian tersebut kondisi jalan tersebut sedang ramai dan juga dipengaruhi oleh kendaraan yang berbelok. Selain itu kendaraan yang saling berdekatan juga mempengaruhi pembacaan prediksi kecepatan kendaraannya.

Selanjutnya pengujian dengan menggunakan Jetson Nano. Pengujian dengan menggunakan modul ini tidak memungkinkan untuk melakukan pengambilan data secara langsung (*real time*) di lokasi. Hal ini karena untuk pengujian dengan menggunakan Jetson Nano terdapat delay dalam pergerakan framenya serta faktor cuaca yang kurang mendukung. Pengujian menggunakan Jetson Nano dilakukan dengan video yang telah direkam sebelumnya. Berikut pada tabel 4.13 merupakan hasil dari pengujian *real time* yang telah dilakukan.

Tabel 4.13 Hasil pengujian menggunakan Jetson Nano

No	Gambar	Jenis Kendaraan	Hasil perhitungan manual (Km/Jam)	Hasil prediksi kecepatan (Km/Jam)	Error Persen (%)
1.	 Gambar 4.45 Hasil pengujian *real time* menggunakan Jetson Nano	Mobil	40,52	36,97	8,76

2.	 Gambar 4.46 Hasil pengujian *real time* menggunakan Jetson Nano	Mobil	39,96	37,89	5,18

3.	 Gambar 4.47 Hasil pengujian *real time* menggunakan Jetson Nano	Motor	44,14	37,3	15,49

4.	 Gambar 4.48 Hasil pengujian *real time* menggunakan Jetson Nano	Motor	42,55	36,4	14,45

5.	 Gambar 4.49 Hasil pengujian *real time* menggunakan Jetson Nano	Mobil	40,74	39,72	2,5

Hasil pengujian yang diperoleh pada tabel 4.13 merupakan hasil pengujian yang didapat dengan menggunakan Jetson Nano sebagai pemrosesan citra digital. Pengujian tersebut menggunakan video berdurasi sekitar 2 menit yang telah diambil di lokasi yang sama dengan pengujian-pengujian sebelumya. Alat- alat yang digunakan diantaranya Jetson Nano, Arduino, LCD , Led Matrix MAX7219, sumber DC (Aki). Adapun hasil yang diperoleh pada pengujian ini didapat nilai error persen tertinggi sebesar 15,49% pada kelas Motor dan diperoleh error persen terendah sebesar 2,5% pada kelas Mobil. Nilai error persen dipengaruhi oleh beberapa faktor diantaranya faktor eksternal (intensitas cahaya, keramaian lalu lintas) dan faktor internal misalnya dari spesifikasi prosesor dan resolusi kamera yang digunakan.

BAB 5. PENUTUP

5.1 Kesimpulan

Berdasarkan pengambilan data yang telah dilakukan pada penelitian ini yang berjudul implementasi algoritma YOLO pada modul kamera untuk deteksi jenis dan kecepatan kendaraan dapat ditarik kesimpulan sebagai berikut.

1. Penggunaan kamera dalam mendeteksi sebuah objek yang bergerak khususnya pada CCTV jalan raya dapat menambahkan sistem yang menggunakan algoritma YOLO sebagai *image processing* dalam membaca kendaraan yang melintas dengan membaginya berdasarkan kelas yang akan dideteksi.
2. Algoritma YOLOV5 yang digunakan pada pendeteksi jenis kendaraan ini berjalan dengan sangat baik dalam mendeteksi kendaraan pada pengujian dengan menggunakan kamera yang beresolusi tinggi (1920 pixel x 1080 pixel). Model yang dipilih pada pengujian ini yaitu YOLOV5s dengan Epoch 300, Batch size 8 dan Optimizer SGD. Model tersebut dipilih karena pada saat pengujian ANOVA, perubahan yang dialami ketika hyperparameter diubah-ubah perubahan output yang diperoleh tidak terlalu signifikan dan untuk bobot yang dihasilkan juga cukup ringan. Selain itu rata-rata error persen yang diperoleh dalam mendeteksi jenis kendaraan sebesar 0% untuk kelas mobil dan 9,91% untuk kelas motor serta error persen pembacaan rata-rata kecepatan kendaraannya diperoleh sebesar 4,45% untuk kelas motor dan 10,22 untuk kelas mobil.
3. Kemampuan sistem dalam mendeteksi jenis kendaraan yang melintas ini sangat dipengaruhi oleh intensitas cahaya. Berdasarkan pengujian yang telah dilakukan, sistem dapat mendeteksi suatu objek pada saat intensitas cahaya berada pada rentang nilai 7000 hingga 15000 lux dengan perolehan rata-rata error persen 17,39% untuk kelas mobil dan 8,19% untuk kelas motor. Hal ini dibuktikan pada saat pengujian deteksi objek dengan nilai intensitas cahaya 205 saat malam hari, sistem tidak dapat membaca objek dengan akurat. Selain itu, dengan tidak

akuratnya pembacaan objek oleh sistem maka hasil prediksi kecepatan kendaraannya juga akan terpengaruh.

5.2 Saran

Mempertimbangkan pengembangan pada penelitian selanjutnya yang terkait dengan penelitian ini, terdapat beberapa saran yang dapat penulis berikan diantaranya:

1. Sistem yang dibuat dalam penelitian ini hanya mendeteksi 2 kelas (Mobil dan Motor), untuk penelitian selanjutnya dapat menambahkan beberapa kelas lainnya seperti truk, bus dan lain sebagainya.
2. Dapat menambahkan deteksi objek dengan dua jalur lintasan kendaraan
3. Dapat menggunakan algoritma YOLO versi terbaru, yang saat ini sudah ada YOLOV7.
4. Sistem pada penelitian ini hanya menguji pada rentang intensitas cahaya 7000 hingga 15000 lux, untuk penelitian selanjutnya dapat menguji pada nilai intensitas yang rendah seperti pada malam hari.
5. Sistem yang dibuat masih terdapat kesalahan dalam mendeteksi objek ketika saat kondisi lalu lintas terlalu ramai. Diharapkan untuk penelitian berikutnya sistem dapat mendeteksi semua kendaraan walaupun dalam keadaan ramai.

DAFTAR PUSTAKA

Andono, P. N., T.Sutojo, dan Muljono. 2017. *PENGOLAHAN CITRA DIGITAL*. PENERBIT ANDI.

Bidang, D., T. Material, J. Teknik, dan M. Universitas. 2018. MESA jurnal fakultas teknik universitas subang. 33–45.

Corporation, N. 2022. Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nano-developer-kit [Diakses pada 6 April 2022].

Davis, E., D. D. Edwards, D. Forsyth, N. J. Hay, J. M. Malik, V. Mittal, M. Sahami, dan S. Thrun. 2010. *Artificial Intelligence A Modern Approach*. *Pearson Education*.

Farnell. 2013. Arduino uno datasheet. *Datasheets*. 1–4.

Feridianto, A. R. 2019. *Rancang Bangun Modul Kamera Deteksi Kecepatan Kendaraan Menggunakan Nvidia Jetson Nano*

Gomaa, A., M. M. Abdelwahab, M. Abo-Zahhad, T. Minematsu, dan R. I. Taniguchi. 2019. Robust vehicle detection and counting algorithm employing a convolution neural network and optical flow. *Sensors (Switzerland)*. 19(20):1–13.

Hamadi, A. M. 2019. Autonomous quadrotor control using convolutional neural networks

Hangzhou Hikvision Digital Technology Co., L. 2022. HIKVISION. https://www.hikvision.com/id/products/IP-Products/Network-Cameras/ [Diakses pada 6 April 2022].

Pramestya, R. H. 2018. Deteksi dan klasifikasi kerusakan jalan aspal menggunakan metode yolo berbasis citra digital. *Institut Teknologi Sepuluh Nopember*. 91.

Redmon, J., S. Divvala, R. Girshick, dan A. Farhadi. 2016. You only look once: unified, real-time object detection joseph. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

Rofii, F., G. Priyandoko, M. I. Fanani, dan A. Suraji. 2021. Vehicle counting accuracy improvement by identity sequences detection based on yolov4 deep neural networks. *Teknik*. 42(2):169–177.

Satura, F. R., A. A. Chandra, dan F. D. Adhinata. 2021. Pengukur kecepatan kendaraan menggunakan algoritma image subtracting. *Journal ICTEE*. 2(2):35.

SrI-VAT, A. 2021. No Title. https://store.arduino.cc/products/arduino-uno-rev3 [Diakses pada 7 Oktober 2022].

Susanto, M. D. 2020. *RANCANG BANGUN MODUL KAMERA PENGUKUR KECEPATAN KENDARAAN MENGGUNAKAN METODE BACKGROUND SUBTRACTION*

Utami, F. R., M. A. Riyadi, dan Y. Christyono. 2020. Sebagai penggerak robot lengan artikulasi. 9(3)

LAMPIRAN

1. Gambar kumpulan dataset kendaraan

Gambar 1 Kumpulan dataset

2. Proses pelabelan menggunakan software LabelImg

Gambar 2 Proses pelabelan objek

3. Proses deteksi kendaraan menggunakan video rekaman CCTV

Gambar 3 deteksi kendaraan menggunakan video rekaman CCTV

4. Proses prediksi kecepatan kendaraan

Gambar 4 Prediksi kecepatan kendaraan

5. Gambar ketika Kendaraan melebihi Kecepatan yang ditentukan

Gambar 5 Hasil tangkapan ketika kendaraan melebihi batas yang ditentukan

6. Pengujian *running text*

Gambar 6 Hasil pengujian *running text*

7. Proses pengujian *real time*

Gambar 7 proses pengujian *real time*

8. Listing program

```
import numpy as np
import cv2
#import torchvision
import torch
import time
#import serial
import joblib
from datetime import datetime

model = torch.hub.load('./', 'custom',
path='best.pt',source='local') # or yolov5m, yolov5l, yolov5x,
custom
cap = cv2.VideoCapture('Pertama.mp4')

#set
j_a = 13
m_a = 55
d_a = 30

prev_frame_time = 0
new_frame_time = 0

#arduino = serial.Serial(port='COM3', baudrate=9600,
timeout=.1)       #'/dev/ttyUSB0'
'''
def write_read(x):
    zz
    time.sleep(0.05)
    data = arduino.readline()
    return data'''

while (cap.isOpened()):

    # Capture frame-by-frame

    ret,frame = cap.read()
    frame = cv2.resize(frame,(800,640),interpolation =
cv2.INTER_AREA)
    if not ret:
        break
    #cv2.line(frame,(800,320),(0,320),(0,255,0),2)
    #cv2.line(frame,(800,400),(0,400),(0,255,0),2)
    gray = frame
    new_frame_time = time.time()
```

```
    fps = 1/(new_frame_time-prev_frame_time)
    prev_frame_time = new_frame_time
    print(abs(fps))
    #s1 = []
    #s2 = []
    detections = model(frame)
    results =
detections.pandas().xyxy[0].to_dict(orient="records")
    for result in results:
                con = result['Confidence']
                cs = result['name']
                x1 = int(result['xmin'])
                y1 = int(result['ymin'])
                x2 = int(result['xmax'])
                y2 = int(result['ymax'])

                now = datetime.now()
                current_time = now.strftime("%H:%M:%S")
                d = now.strftime("%S")
                h = int( datetime.now().strftime("%H"))
                m = int( datetime.now().strftime("%M"))
                s = int( datetime.now().strftime("%S"))
                h1 = h-j_a
                m1 = m-m_a
                s1 = s-d_a
                sv = [h,m,s,h1,m1,s1]

                wx =(x2+x1)//2
                wy = (y2+y1)//2
                cent = (wx,wy)
                print(wy)
                #cv2.line(frame,(800,400),(0,400),(0,255,0),2)
                #cv2.line(frame,(800,440),(0,440),(0,255,0),2)
                g1=400
                g2=440
                #cv2.rectangle(frame, (x1, y1), (x2, y2),(0, 255,
255), 2)
                #cv2.circle(frame,(cent) ,radius=2,color=(0, 0,
255),thickness= 2)
                #s1.append(tot)
                #joblib.dump(tot,'simpen.pkl')
                #print(tot)

                if cent[1] > g1 and cent[1] < 420  :
                    xx1 = fps
```

```
                    joblib.dump(xx1,'data1.pkl')

                if cent[1] > 420 and cent[1] < g2  :
                    xx2 = fps
                    joblib.dump(xx2,'data2.pkl')

                #if cent[1]>400 :

                #print(joblib.load('data1.pkl'),joblib.load('data2
.pkl'))

                if con> 0.5 :

                    if cs == "motor" :
                                cv2.rectangle(frame, (x1, y1), (x2,
y2),(0, 255, 255), 2)
                                cv2.circle(frame,(cent)
,radius=2,color=(0, 0, 255),thickness= 2)
                                cv2.putText(frame,str(cs)+" ", (x1,y1
+ 30), cv2.FONT_HERSHEY_PLAIN,2, (0,255,2),2)
                                if cent[1]> g1 and cent[1]<g2:
                                    d1 =joblib.load('data1.pkl')
                                    d2 =joblib.load('data2.pkl')
                                    tot = (d1+d2)
                                    S=tot/28
                                    Speed = (6/1000)/(S/3600)
                                    cv2.putText(frame,str(round((Speed
), 2))+"km/h", (x1,y1 -20), cv2.FONT_HERSHEY_PLAIN,2, (0,0,255),2)
                                    #arduino.write(bytes(str(round(((5
.5/1000)/S*3600), 2)), 'utf-8'))
                                    #cv2.putText(frame,str(round(tot*7
50/25, 2))+"km/h", (x1,y1 -30), cv2.FONT_HERSHEY_PLAIN,2,
(0,0,255),2)

                                    if Speed > 50 :
                                        img = frame[y1:y2, x1:x2]
                                        cv2.imshow('pelanggaran', img)
                                        #img = cv2.resize(img, None,
fx=1, fy=1,interpolation=cv2.INTER_CUBIC)
                                        cv2.imwrite(f"captures/{str(sv
)}.jpg", img)
                                #cv2.putText(frame,str(cs)+" ", (x1,y1
+ 30), cv2.FONT_HERSHEY_PLAIN,2, (0,255,255),2)
                                #cv2.putText(frame,str(round(con,2))+"
", (x1-20,y1 + 50), cv2.FONT_HERSHEY_PLAIN,2, (0,255,255),2)
```

```
                            #cv2.putText(frame,(i)+" ", (x1-20,y1
+ 50), cv2.FONT_HERSHEY_PLAIN,2, (0,255,255),2)

                    else :
                            cv2.rectangle(frame, (x1, y1), (x2,
y2),(0, 255, 0), 2)
                            cv2.circle(frame,(cent)
,radius=2,color=(0, 0, 255),thickness= 2)
                            cv2.putText(frame,str(cs)+" ", (x1,y1
+ 30), cv2.FONT_HERSHEY_PLAIN,2, (0,255,2),2)
                            if cent[1]> g1 and cent[1]<g2:
                                d1 =joblib.load('data1.pkl')
                                d2 =joblib.load('data2.pkl')
                                tot = (d1+d2)
                                S=tot/28
                                Speed = (6/1000)/(S/3600)
                                cv2.putText(frame,str(round((Speed
), 2))+"km/h", (x1,y1 -20), cv2.FONT_HERSHEY_PLAIN,2, (0,0,255),2)
                                #arduino.write(bytes(str(round(((5
.5/1000)/S*3600), 2)), 'utf-8'))
                                #cv2.putText(frame,str(round(tot*6
00/25, 2))+"km/h", (x1,y1 -30), cv2.FONT_HERSHEY_PLAIN,2,
(0,0,255),2)
                                if Speed > 50 :
                                    img = frame[y1:y2, x1:x2]
                                    cv2.imshow('pelanggaran', img)
                                    #img = cv2.resize(img, None,
fx=1, fy=1,interpolation=cv2.INTER_CUBIC)
                                    cv2.imwrite(f"captures/{str(sv
)}.jpg", img)
                            #cv2.putText(frame,str(round(con,2))+"
", (x1-20,y1 + 50), cv2.FONT_HERSHEY_PLAIN,2, (0,255,255),2)
                            #cv2.putText(frame,(i)+" ", (x1-20,y1
+ 50), cv2.FONT_HERSHEY_PLAIN,2, (0,255,255),2)

    cv2.imshow('frame',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

9. Linsting program Arduino

```
// Including the required Arduino libraries
#include <MD_Parola.h>
```

```
#include <MD_MAX72xx.h>
#include <SPI.h>

// Uncomment according to your hardware type
#define HARDWARE_TYPE MD_MAX72XX::FC16_HW
//#define HARDWARE_TYPE MD_MAX72XX::GENERIC_HW

// Defining size, and output pins
#define MAX_DEVICES 4
#define CS_PIN 3

// Create a new instance of the MD_Parola class with hardware SPI connection
MD_Parola myDisplay = MD_Parola(HARDWARE_TYPE, CS_PIN,
MAX_DEVICES);

textEffect_t texteffect[] =
{
  PA_SCROLL_LEFT
};

void setup() {
  // Intialize the object
  myDisplay.begin();
  Serial.begin(9600);
  // Set the intensity (brightness) of the display (0-15)
  myDisplay.setIntensity(0);

  // Clear the display
  myDisplay.displayClear();
}

void loop() {
 while (Serial.available() == 0) {}     //wait for data available
  String teststr = Serial.readString();  //read until timeout
  teststr.trim();                        // remove any \r \n whitespace at the end of the
String

  myDisplay.setTextAlignment(PA_LEFT);
  myDisplay.print( teststr);
  delay(500);
  myDisplay.displayClear();
  delay(500);
  myDisplay.print( teststr);
  delay(500);
}
```