APLIKASI PENGHITUNG KENDARAAN YANG MELINTAS DI JALAN RAYA BERDASARKAN METODE YOLO OBJECT DETECTION

TUGAS AKHIR

Program Studi

S1 TEKNIK KOMPUTER

Oleh:

Rezza Ifmanur Isnaini

15410200068

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2020

APLIKASI PENGHITUNG KENDARAAN YANG MELINTAS DI JALAN RAYA BERDASARKAN METODE YOLO OBJECT DETECTION

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Sarjana Teknik

UNIVERSITAS Dinamika

Oleh :

Nama : Rezza Ifmanur Isnaini

NIM : 15410200068

Program Studi : S1 Teknik Komputer

FAKULTAS TEKNOLOGI DAN INFORMATIKA
UNIVERSITAS DINAMIKA
2020

TUGAS AKHIR

APLIKASI PENGHITUNG KENDARAAN YANG MELINTAS DI JALAN RAYA BERDASARKAN METODE YOLO OBJECT DETECTION

Dipersiapkan dan disusun oleh

Rezza Ifmanur Isnaini

NIM : 15410200068

Telah diperiksa, diuji dan disetujui oleh Dewan Pembahas

Pada : 21 Agustus 2020

Susunan Dewan Pembahas

Pembimbing

I. Dr. Susijanto Tri Rasmana, S.Kom., M.T.
NIDN. 0727097302

Digitally signed by Universitas Dinamika
Date: 2020.08.31 12:11:26 +07'00'

II. Pauladie Susanto, S.Kom., M.T.
NIDN. 0729047501

Digitally signed by Universitas Dinamika
Date: 2020.09.01 12:15:33 +07'00'

Pembahas

I. Weny Indah Kusumawati. S.Kom., M.MT.
NIDN. 0721047201

Digitally signed by Universitas Dinamika
Date: 2020.09.01 12:46:00 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

Untuk memperoleh gelar Sarjana

Dr. Jusak

Digitally signed by Universitas Dinamika
Date: 2020.09.02 10:40:20 +07'00'

NIDN : 0708017101

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

UNIVERSITAS Dinamika

“Jangan Lupa Bersyukur”

UNIVERSITAS Dinamika

Kupersembahkan Kepada ALLAH SWT

Ibu, Bapak dan semua keluarga tercinta,

Yang selalu mendukung, memotivasi dan menyisipkan nama saya dalam doa-doa terbaiknya.

Beserta semua teman yang selalu membantu, mendukung dan memotivasi agar tetap berusaha menjadi lebih baik

UNIVERSITAS Dinamika

SURAT PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa Universitas Dinamika, saya :

Nama	: Rezza Ifmanur Isnaini
NIM	: 15410200068
Program Studi	: S1 Teknik Komputer
Fakultas	: Fakultas Teknologi dan Informatika
Jenis Karya	: Tugas Akhir
Judul Karya	: **APLIKASI PENGHITUNG KENDARAAN YANG MELINTAS DI JALAN RAYA BERDASARKAN METODE YOLO OBJECT DETECTION**

Menyatakan dengan sesungguhnya bahwa:

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, saya menyetujui memberikan kepada Universitas Dinamika Hak Bebas Royalti Non-Eksklusif (*Non- Exclusive Royalti Free Right)* atas seluruh isi/ sebagian karya ilmiah saya tersebut di atas untuk disimpan, dialihmediakan dan dikelola dalam bentuk pangkalan data *(database)* untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta
2. Karya tersebut di atas adalah karya asli saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya atau pendapat orang lain yang ada dalam karya ilmiah ini adalah semata hanya rujukan yang dicantumkan dalam Daftar Pustaka saya
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiat pada karya ilmiah ini, maka saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada saya.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Surabaya, 8 September 2020
Yang menyatakan

Rezza Ifmanur Isnaini
Nim : 15410200068

UNIVERSITAS Dinamika

ABSTRAK

Berbagai macam dampak yang dihasilkan oleh kemacetan dan bersifat negatif, setelah dilihat dari berbagai aspek. Kemacetan menimbulkan banyak kerugian dari berbagai segi materi, waktu, dan tenaga. Seperti dari aspek ekonomi kemacetan menghambat proses produksi dan distribusi sehingga menghambat laju perekonomian yang harusnya berjalan. Dari aspek Kesehatan kemacetan menyebabkan dampak negatif yaitu mempengaruhi kondisi fisik dan psikis para pengguna jalan raya. Sudah banyak upaya yang dilakukan untuk megurangi kepadatan kendaraan di jalan raya. Dengan cara membuat sebuah sistem penghitung jumlah kendaraan dengan menggunakan CCTV yang ada pada salah satu lampu lalu lintas di Surabaya dilakukan secara otomatis dan diproses pada pengolahan citra digital menggunakan metode YOLO *object detection*. Hasil pengujian yang didapatkan aplikasi YOLO *object detection* yang sudah bisa mendeteksi dan membedakan kendaraan dengan unsur lain. Dengan pendeteksian kali ini akan lebih mudah untuk dapat menghitung kendaraan yang lewat pada jalan raya di video. Setelah itu YOLO *object detection* dapat menghitung kendaraan ditandai dengan adanya tampilan di layer pada bagian pojok kanan. Kemudian dengan pengujian menggunakan 3 video yang diambil dari hasil rekam dan dari internet yang sudah bisa mendeteksi dan menghitung dengan akurat yang mempunyai nilai keakuratan paling tinggi 56% dan paling rendah 10%.

Kata Kunci : *Kemacetan, Kendaraan, Kerugian, YOLO object Detection*

UNIVERSITAS Dinamika

KATA PENGANTAR

Puji Syukur penulis panjatkan kepada ALLAH SWT, karena dengan rahmat dan hidayah-Nya, penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul "APLIKASI PENGHITUNG KENDARAAN YANG MELINTAS DI JALAN RAYA BERDASARKAN METODE YOLO OBJECT DETECION". Laporan Tugas Akhir ini diajukan dan dibuat untuk memenuhi syarat guna mendapatkan gelar sarjana dari pada Fakultas Teknologi dan Informatika di Universitas Dinamika Surabaya.

Selama membuat laporan Tugas Akhir ini, penulis banyak menerima bantuan dan dukungan sehingga dapat menyelesaikan laporan Tugas Akhir ini. Oleh karena itu, penulis mengucapkan terima kasih sebesar-besarnya kepada:

1. Kedua orang tua dan saudara-saudara tercinta atas kasih sayang yang tak terbatas, perjuangan tiada henti dan doa yang dipanjatkan setiap waktu untuk penulis.
2. Bapak Dr. Jusak selaku Dekan Fakultas Teknologi dan Informatika (FTI) Universitas Dinamika Surabaya.
3. Bapak Pauladie Susanto, S.Kom., M.T., selaku Ketua Program Studi S1 Teknik Komputer Universitas Dinamika Surabaya.
4. Bapak Dr. Susijanto Tri Rasmana, S.Kom., M.T., dan Bapak Pauladie Susanto, S.Kom., M.T., selaku dosen Pembimbing yang selalu memberi bimbingan dalam menyelesaikan Tugas Akhir dan laporan ini.
5. Ibu Weny Indah Kusumawati, S.Kom., M.MT., selaku Dosen Pembahas atas masukan dalam menyusun Tugas Akhir ini.
6. Semua Staff Dosen yang telah membagi ilmunya.
7. Teman Teknik Komputer dari berbagai Angkatan dan semua pihak yang terlibat. Terima kasih karena sudah membantu dan menemani hingga dapat menyelesaikan Tugas Akhir dan laporan ini.
8. Serta semua pihak yang tidak bisa disebutkan yang sudah membantu menyelesaikan Tugas Akhir ini secara tidak langsung maupun secara langsung.

Penulis menyadari bahwa laporan Tugas Akhir ini masih jauh dari sempurna karena adanya keterbatasan ilmu dan pengalaman yang dimiliki. Oleh karena itu, semua kritik dan saran yang bersifat membangun akan penulis terima dengan senang hati. Penulis berharap, semoga skripsi ini dapat bermanfaat bagi semua pihak yang memerlukan.

Surabaya, 21 Agustus 2020

Rezza Ifmanur Isnaini

UNIVERSITAS Dinamika

DAFTAR ISI

UNIVERSITAS Dinamika

UNIVERSITAS Dinamika

DAFTAR GAMBAR

UNIVERSITAS Dinamika

DAFTAR TABEL

UNIVERSITAS Dinamika

DAFTAR LAMPIRAN

UNIVERSITAS Dinamika

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kendaraan merupakan alat yang digunakan hampir semua manusia untuk bermobilitas atau berpindah dari tempat yang jauh ataupun dekat. Terdapat berbagai jenis kendaraan (sepeda motor) dan beroda empat (mobil, truk, dan bus). Kemajuan teknologi transportasi berdampak pada perkembangan lalu lintas dan angkutan jalan, sehingga terjadi perubahan pada prasarana jalan, sarana angkutan, dan perangkat lalu lintas lain-lain. Faktor yang lainnya ialah pertumbuhan ekonomi yang menyebabkan pengguna jalan semakin meningkat dan semakin padat di jalan raya. Indonesia merupakan salah satu negara dengan tingkat pembelian kendaraan bermotor yang tinggi.

Berbagai macam dampak yang dihasilkan oleh kemacetan dan bersifat negatif, setelah dilihat dari berbagai aspek, kemacetan menimbulkan banyak kerugian dari berbagai segi yaitu materi, waktu, dan tenaga. Seperti dari aspek ekonomi kemacetan menghambat proses produksi dan distribusi sehingga menghambat laju perekonomian yang harusnya berjalan. Dari aspek Kesehatan kemacetan menyebabkan dampak negatif yaitu mempengaruhi kondisi fisik dan psikis para pengguna jalan raya, terebih bagi mereka yang sering beraktivitas di jalan saat berangkat dan pulang dari bekerja, dan lain sebagainya. Terjadinya kemacetan adalah dampak dari ketidakseimbangan jaringan lalu lintas karena adanya penumpukan kendaraan yang mengakibatkan kepadatan lalu lintas pada suatu jalan menjadi tinggi sehingga arus lalu lintas menjadi tersendat bahkan berhenti.

Dikarenakan permasalahan tersebut maka di dalam Tugas Akhir ini dilakukan penelitian untuk mengurangi kepadatan kendaraan di jalan raya. Salah satunya pada penelitian ini dibuat sebuah sistem penghitung jumlah kendaraan dengan menggunakan CCTV yang ada pada salah satu lampu lalu lintas di Surabaya dilakukan secara otomatis dan diproses pada pengolahan citra digital menggunakan

metode YOLO *Object Detection*. Algoritma YOLO ini merupakan *real object detection*, dengan kemampuan pada *base* model yang bisa memproses 45 *frame per second* secara *real-time* dan pada versi yang lebih kecil bisa memproses 155 *frame per second* secara *real-time*.

Sudah banyak upaya yang dilakukan pada penelitian lain yang mengarah pada permasalahan yang hampir sama. Salah satunya adalah sebagai berikut, (Karlina dan Indarti, 2019) dengan judul pengenalan objek makanan cepat saji pada video dan real time webcam menggunakan metode you only look once (YOLO). Didalam penelitian ini membahas tentang keakuratan metode yolo yang digunakan untuk mendeteksi makanan cepat saji menggunakan video maupun secara real time menggunakan webcam.

Selain itu penelitian lainnya yang menggunakan kendaraan sebagai objek pengolahan citra adalah kendali lampu lalu lintas dengan deteksi kendaraan menggunakan metode blob detection oleh (Qory Hidayati, 2017). Didalam penelitian ini membahas tentang metode selain YOLO yang bisa digunakan untuk mendeteksi kendaraan menggunakan video ataupun secara real time.

UNIVERSITAS Dinamika

1.2 Perumusan Masalah

Berdasarkan latar belakang diatas maka dirumuskan permasalahan:

1. Bagaimana cara membuat aplikasi pendeteksi kendaraan dengan rekaman pada salah satu ruas jalan?
2. Bagaimana cara menghitung kendaraan dengan aplikasi metode YOLO *object detection*?

1.3 Batasan Masalah

Untuk menghindari pembahasan yang lebih luas terkait dengan perancangan aplikasi penghitug kendaraan meggunakan CCTV di lampu lalu lintas, maka penelitian ini ditentukan pada ruang lingkup ternetu antara lain :

1. CCTV lampu lalu lintas yang tidak disiarkan langsung (*real time*) karena hanya menggunakan rekaman video saja.

2. Cuaca di daerah CCTV lampu lalu lintas cerah, tidak berkabut, tidak gerimis, dan tidak hujan.
3. Aplikasi hanya bisa digunakan pada intensitas cahaya yang cukup.
4. Menghitung kendaraan pada satu ruas jalan yang disorot CCTV lampu lalu lintas.
5. Kendaraan yang dihitung program hanya kendaraan yang memiliki roda 4 dan lebih.

1.4 Tujuan

Berdasarkan rumusan masalah yang diuraikan diatas, maka tujuan penelitian ini adalah sebagai berikut:

1. Melakukan penerapan program pengolahan citra digital menggunakan metode YOLO object detection untuk mendeteksi kendaraan pada salah satu ruas jalan.
2. Merancang aplikasi penghitung kendaraan dari metode YOLO object detection pada salah satu ruas jalan menggunakan rekaman pada CCTV di lampu lalu lintas.

UNIVERSITAS Dinamika

1.5 Sistematika Penulisan

Untuk mempermudah pembaca dalam memahami persoalan dan pembahasan, maka penulisan laporan Tugas Akhir ini dibuat dengan sistematika sebagai berikut.

BAB I PENDAHULUAN

Pada bab I membahas tentang latar belakang masalah dan penjelasan permasalahan secara umum, perumusan masalah serta Batasan masalah yang dibuat, tujuan dari pembuatan Tugas Akhir dan sistematika penulisan buku.

BAB II LANDASAN TEORI

Pada bab II membahas teori – teori yang berhubungan dan mendukung dalam pembuatan Tugas Akhir seperti kendaraan, YOLO object detecton, deep learning, komputer vision, dan literatur yang memunjang dalam pembuatan Tugas Akhir ini.

BAB III ANALISIS DAN PERANCANGAN SISTEM

Pada bab III membahas tentang perancangan sistem aplikasi pada bagian program sistem, maupun model sistem aplikasi.

BAB IV PENGUJIAN

Pada bab IV menjelaskan tentang hasil pengujian aplikasi, pengujian yang dilakukan yaitu uji deteksi kendaraan, dan uji aplikasi penghitung kendaraan.

BAB V PENUTUP

Pada bab V menjelaskan tentang kesmipulan dan saran. Kesimpulan berdasarkan hasil penelitian dari mulai pengerjaan Tugas Akhir, serta saran yang perlu diperbaiki untuk penegembang.

UNIVERSITAS Dinamika

BAB II
LANDASAN TEORI

2.1 Kendaraan

Kendaraan adalah alat transportasi yang digerakan oleh mesin ataupun makhluk hidup. Arti kata kendaraan menurut KBBI Nomina (kata benda) sesuatu yang digunakan untuk dikendarai atau dinaiki (seperti kuda, mobil, kereta). Kendaraan biasanya digunakan oleh manusia untuk bermobilitas atau berpindah tempat dari yang dekat maupun jauh. Pada umumnya kendaraan dibuat oleh manusia contohnya motor, mobil, perahu, dan pesawat. Kendaraan tak bermotor bisa juga digerakan oleh manusia atau ditarik oleh hewan, seperti gerobak, kendaraan tertua yang pernah diciptakan manusia adalah perahu.

2.2 Training Model

Training model adalah analisa statistik untuk membantu komputer membuat keputusan dan prediksi berdasarkan karakteristik yang ditemukan dalam data tersebut. Dengan kata lain, itu adalah tindakan memiliki komputer mengurai aliran data untuk membentuk pemahaman abstrak tentang hal itu (disebut "model"), dan menggunakan model itu untuk membandingkannya dengan data yang lebih baru. contohnya adalah ketika Anda mengetik pesan, koreksi otomatis memprediksi apa yang akan Anda ketik selanjutnya menggunakan model Machine Learning yang terus diperbarui saat Anda mengetik. Bahkan asisten virtual, seperti Siri, Alexa, dan Google Assistant sepenuhnya bergantung pada Machine Learning untuk meniru perilaku mirip manusia. Berikut ini urutan cara yang digunakan untuk bisa menerapkan training objek yang benar:

2.2.1 Kumpulan Data Dan Gambar

Mengklasifikasikan gambar-gambar yang akan digunakan untuk pendeteksian. Yaitu dengan cara menemukan gambar-gambar yang memiliki objek

utama yang berbeda untuk menghindari membingungkan model. Lalu membagi kumpulan data menjadi dua kategori: melatih model, dan untuk pengujian.

2.2.2 Melatih Model

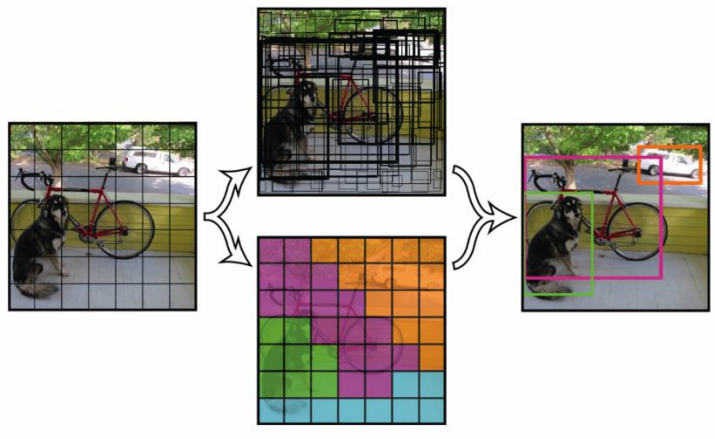
Menulis beberapa kode dan menempatkan aplikasi Swift dan Xcode untuk melakukan pelatihan model.

2.2.3 Menguji dan Mengekspor

Menguji hasil training langsung di aplikasi tanpa harus membuat proyek. Ketika model sudah cukup akurat, ekspor dalam format Core ML dan menggunakannya di aplikasi pendeteksi.

2.3 YOLO Object Detection

YOLO (*you only look once*) merupakan algoritma *real object detection* yang baru–baru ini sangat popular untuk dikembangkan. YOLO menggunakan pendekatan yang sangat berbeda dengan algoritma sebelumnya, yakni menerapkan jaringan syaraf tunggal pada keseluruhan gambar. Pendeteksian objek dilakukan dengan membingkai objek yang akan dideteksi sebagai *regression problem* dan memisahkan special pada *bounding boxes* dan *class probabilities*. Dengan menggunakan *single neural network* untuk memprediksi *bounding boxes* dan *class probabilities* dari seluruh gambar pada satu kali evaluasi. Karena metode ini menggunakan *single neural network* untuk semua *detection pipeline*, maka permforma deteksi ini bisa dioptimasi dari *end-to-end* (Redmon, Diyyala, Girshick, dan Farhadi, 2015).



Gambar 2. 1 Klasifikasi YOLO Object Detection
(Sumber: https://machinelearning.mipa.ugm.ac.id/2018/08/05/yolo-you-only-look-once/)

YOLO memiliki kemampuan pada base model yang bisa memproses *45 frame per second* secara *real time* pada versi yang lebih kecil bisa memproses *155 frame per second* secara *real-time*. YOLO juga memiliki arsitektur yang sederhana yaitu jaringan syaraf convolutional.

Layer	kernel	stride	output shape
Input			(416, 416, 3)
Convolution	3x3	1	(416, 416, 16)
MaxPooling	2x2	2	(208, 208, 16)
Convolution	3x3	1	(208, 208, 32)
MaxPooling	2x2	2	(104, 104, 32)
Convolution	3x3	1	(104, 104, 64)
MaxPooling	2x2	2	(52, 52, 64)
Convolution	3x3	1	(52, 52, 128)
MaxPooling	2x2	2	(26, 26, 128)
Convolution	3x3	1	(26, 26, 256)
MaxPooling	2x2	2	(13, 13, 256)
Convolution	3x3	1	(13, 13, 512)
MaxPooling	2x2	1	(13, 13, 512)
Convolution	3x3	1	(13, 13, 1024)
Convolution	3x3	1	(13, 13, 1024)
Convolution	1x1	1	(13, 13, 125)

Gambar 2. 2 Jaringan Saraf Convolutional
(Sumber: https://machinelearning.mipa.ugm.ac.id/2018/08/05/yolo-you-only-look-once/)

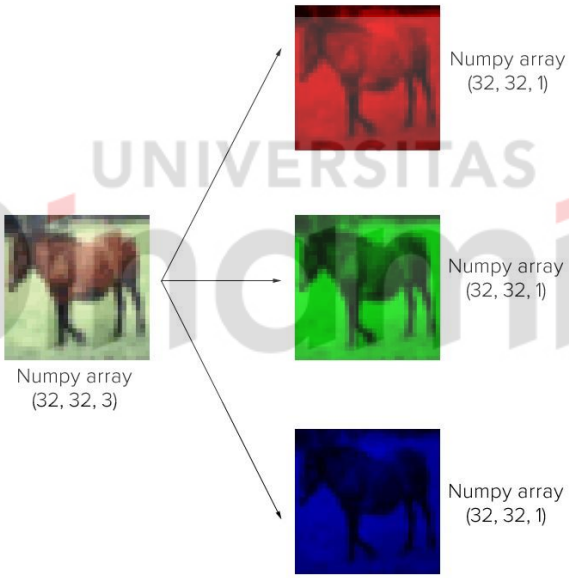
Jaringan saraf ini hanya menggunakan jenis lapisan standar; konvolusi dengan kernel 3 x 3 dan *max pooling* dengan 2 x 2 kernel. Lapisan konvolusional terkahir 1 x 1 kernel digunakan untuk mengecilkan data ke bentuk 13 x 13 x 125.

13 x 13 ini seharusnya terlibat familiar: itu adalah ukuran *grid* yang dibagi menjadi gambar. 125 merupakan *channel* untuk setiap *grid*. 125 ini berisi data untuk kotak pembatas dan prediksi kelas. Setiap sel *grid* memprediksi 5 kotak sekeliling dan dijelakan oleh 25 elemen data.

- X, Y untuk lebar dan tinggi kotak pembatas (dua elemen data)
- Skor keyakinan (satu elemem data)
- Distribusi probabilitas yang lebih dari 20 kelas (20 elemen data)

Berikut ini urutan rumus atau cara yang digunakan untuk bisa menerapkan metode YOLO *object detection*.

2.3.1 Convolutional Layer



Gambar 2. 3 Pemisahan RGB pada Gambar
(Sumber: https://medium.com/@16611110/apa-itu-convolutional-neural-network-836f70b193a4)

Gambar diatas adalah RGB (*Red, Green, Blue*) *image* berukuran 32 x 32 pixels yang sebenarnya adalah multidimensional array dengan ukuran 32 x 32 x 3 (3 adalah jumlah *channel*). Convolutional layer terdiri dari neuron yang tersusun sedemikiran rupa sehingga membentuk sebuah filter dengan Panjang dan tinggi (*pixels*). Sebagai contoh, layer pertama pada *feature extraction layer* biasanya

adalah *convolutional layer* dengan ukuran 5 x 5 x 3. Panjang 5 *pixels*, tinggi 5 *pixels*, dan tebal/jumlah 3 buah sesaui dengan *channel* dari gambar tersebut.

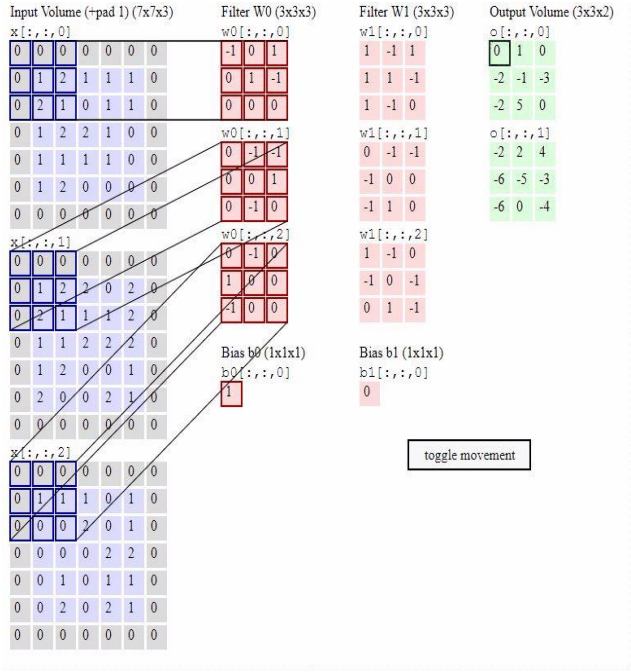
Untuk menghitung dimensi dari *feature map* bisa menggunakan rumus seperti dibawah ini:

$$Output = \frac{W - N + 2P}{S} + 1$$

- W = Panjang/Tinggi Input
- N = Panjang/Tinggi Filter
- P = Padding
- S = Stride

Stride adalah parameter yang menyatukan beberapa jumlah pergeseran filter. Jika nilai stride adalah 1, maka convolutional filter akan bergeser sebanyak 1 pixels secara horizontal lalu vertical. Semakin kecil stride maka akan semakin detail informasi yang kita dapatkan dari sebuah input, namun membutuhkan komputasi yang lebih jika dibandingkan dengan stride yang besar.

Padding atau *zero padding* adalah parameter yang membutuhkan jumlah *pixels* (berisi nilai 0) yang akan ditambahkan di setiap sisi input. Hal ini digunakan dengan tujuan untuk memanipulasi dimensi output dari *convolutional layer* (feature map). Dimensi output dari convolutional layer selalu lebih kecil dari inputnya (keculai penggunaan 1 x 1 filter dengan stride 1). Padding dapat mengatur dimensi output agar tetap sama seperti dimensi input atau setidaknya tidak berkurang secara *drastic*.



Gambar 2. 4 Proses *Convolutional Layer*
(Sumber: https://medium.com/@16611110/apa-itu-convolutional-neural-network-836f70b193a4)

Untuk menghitung output konvolusi menggunakan rumus dibawah ini:

$$Konvolusi = \sum_{i=0}^{2} \sum_{j=0}^{2} Ii,j * Fi,j$$

- **Tahap 1**

Channel 1									
Input pad.1				Filter W0				Konvolusi	
0	0	0		-1	0	1		0	0	0
0	1	2	x	0	1	-1	=	0	1	-2
0	2	1		0	0	0		0	0	0

Konvolusi = $I_{0,0}$ X $F_{0,0}$ + $I_{0,1}$ X $F_{0,1}$ + $I_{0,2}$ X $F_{0,2}$ + $I_{1,0}$ X $F_{1,0}$ + $I_{1,1}$ X $F_{1,1}$ + $I_{1,2}$ X $F_{1,2}$ + $I_{2,0}$ X $F_{2,0}$ + $I_{2,1}$ X $F_{2,1}$ + $I_{2,2}$ X $F_{2,2}$

Konvolusi = 0 + 0 + 0 + 0 + 1 + -2 + 0 + 0 + 0

Konvolusi = -1

Channel 2									
Input Pad.1				Filter W0				Konvolusi	
0	0	0		0	-1	-1		0	0	0
0	1	2	x	0	0	1	=	0	0	2
0	2	1		0	-1	0		0	-2	0

Konvolusi = $I_{0,0}$ X $F_{0,0}$ + $I_{0,1}$ X $F_{0,1}$ + $I_{0,2}$ X $F_{0,2}$ + $I_{1,0}$ X $F_{1,0}$ + $I_{1,1}$ X $F_{1,1}$ + $I_{1,2}$ X $F_{1,2}$ + $I_{2,0}$ X $F_{2,0}$ + $I_{2,1}$ X $F_{2,1}$ + $I_{2,2}$ X $F_{2,2}$

Konvolusi = 0 + 0 + 0 + 0 + 0 + 2 + 0 + -2 + 0

Konvolusi = 0

Channel 3										
Input Pad.1				Filter W0				Konvolusi		
0	0	0		0	-1	0		0	0	0
0	1	1	x	1	0	0	=	0	0	0
0	0	0		-1	0	0		0	0	0

Konvolusi = $I_{0,0}$ X $F_{0,0}$ + $I_{0,1}$ X $F_{0,1}$ + $I_{0,2}$ X $F_{0,2}$ + $I_{1,0}$ X $F_{1,0}$ + $I_{1,1}$ X $F_{1,1}$ + $I_{1,2}$ X $F_{1,2}$ + $I_{2,0}$ X $F_{2,0}$ + $I_{2,1}$ X $F_{2,1}$ + $I_{2,2}$ X $F_{2,2}$

Konvolusi = 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0

Konvolusi = 0

- **Tahap 2**

Output Konvolusi = Konvolusi Ch.1 + Kovolusi Ch.2 + Konvolusi Ch.3 + Bias b1

Output Konvolusi = -1 + 0 + 0 + 1

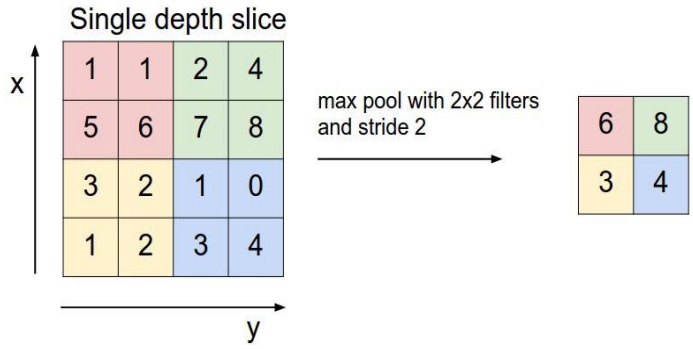
Output Konvolusi = 0

Output Konvolusi		
0		

2.3.2 Pooling Layer

Pooling layer biasanya berbeda setelah convolutional layer. Pada prinsipnya *pooling layer* terdiri dari sebuah *filter* dengan ukuran dan *stride* tertentu yang akan bergeser pada seluruh area *feature map*.

UNIVERSITAS Dinamika



Gambar 2. 5 Proses *Pooling Layer* dengan cara *Max Pool*
(Sumber: https://medium.com/@16611110/apa-itu-convolutional-neural-network-836f70b193a4)

Dimensi *output* dari *pooling layer* juga menggunakan rumus yang sama seperti pada *convolutional layer*. Dengan tujuan adalah mengurangi dimensi dari *feature map* (*Downsampling*), sehingga mempercepat komputasi karena parameter yang harus diperbarui semakin sedikit dan mengatasi *overfitting*.

2.3.3 Fully Connected Layer (FC Layer)

Feature map yang dihasilkan dari *feature extraction layer* masih berbentuk *multidimensional array*, sehingga harus melakukan "*flatten*" atau *reshape feature map* menjadi sebuah vektor agar bisa kita gunakan sebgaai *input* dari *fully-commected layer.*

2.4 Deep Learning

Deep structured learning adalah salat satu cabang dari ilmu pembelajaran mesin (*Machine Learning*) yang terdiri algoritma pemodelan abstarksi tingkat tinggi pada data menggunakan sekumpulan fungsi transformasi *non-linear* yang ditata berlapis-lapis dan mendalam. Teknik dan algoritma dalam pembelajaran dalam dapat digunakan baik untuk kebutuhan pembelajaran terarah (*supervised learning*), pembelajaran tak terarah (*unsupervised learning*) dan semi terarah (*semi-supervised learning*) dalam bebagai aplikasi seperti pengenalan citra, pengenalan suara, klasifikasi teks, dan sebagainya. *Deep leraning* disebut sebagai *deep* (dalam)

karena struktur dan jumlah jaringan saraf pada algoritmanya sangat banyak bisa mencapai hingga ratusan lapisan.

Deep learning adalah salah satu jenis algoritma jaringan saraf tiruan yang menggunakan metadata sebagai input dan mengolahnya menggunakan sejumlah lapisan tersembunyi (*hidden layer*) transformasi non linear dari data masukan untuk menghitung nilai output. Algoritma pada *deep learning* memiliki fitur yang unik yaitu sebuah fitur yang mampu mengekstraksi secara otomatis. Hal ini berarti algoritma yang dimilikinya secara otomatis dapat menangkap fitur yang relevan sebagai keperluan dalam pemecahan suatu masalah. Algoritma semacam ini sangat penting dalam sebuah kecerdasan buatan karena mampu mengurangi beban pemrograman dalam memilih fitur yang eksplisit. Dan, algoritma ini dapat digunakan untuk memecahkan permasalahan yang perlu pengawasan (*supervised*), tanpa pengawasan (*unsupervised*), dan semi terawasi (*semi supervised*).

Jenis *Deep Learning* :

- *Deep Learning* untuk pembelajaran tanpa pengawasan (*Unsupervised Learning*): tipe ini digunakan pada saat tabel variabel target tidak tersedia dan korelasi nilai yang lebih tinggi harus dihitung dari unit yang diamati untuk menganalisa polanya.
- *Hybrid Deep Networks* (*Deep Learning* Gabungan): pendekatan tipe ini bertujuan agar dapat dicapai hasil yang baik dengan menggunakan pembelajaran yang diawasi untuk melakukan Analisa pola atau dapat juga dengan menggunakan pembelajaran tanpa pengawasan.

2.5 Komputer Vision

Komputer vision merupakan bidang yang mencakup metode untuk memperoleh, mengolah, menganalisis, dan memahami data visual seperti gambar dan video. Tujuan utamanya adalah agar komputer atau mesin dapat meniru kemampuan memahami dan menginterprestasikan informasi sensorik atau kemampuan intelektual untuk mencari makna yang diterima oleh panca indera mata manusia dan otak.

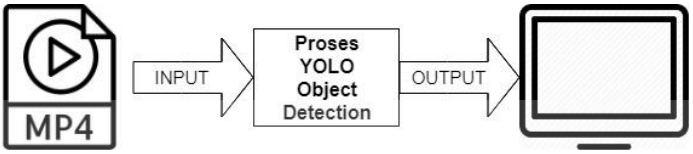
Bidang yang berkaitan erat dengan *komputer vision* adalah *image processing* (pengolahan citra) dan machine vision (visi mesin). Ada tumpeng tindih yang signifikan dalam berbagai terknik dan aplikasi yang mencakup tiga bidang tersebut. Hal ini menunjukan terknik dasar yang diguakan dan dikembangkan kurang lebih sama (*identic*). Komputer vision mencakup teknologi utama untuk menganalisis citra (visual) secara otomatis yang digunakan dalam bidang lain. Sedangkan machine vision biasanya mengacu pada proses menggabungkan analisis citra otomatis dengan metode atau teknologi lain baik berupa *software* maupun *hardware* untuk mencapai tujuan tertentu.

UNIVERSITAS Dinamika

BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian

Metode Penelitian Tugas Akhir ini adalah impelementasi algoritma YOLO *object detection* pada kendaraan yang digunakan untuk menghitung kendaraan yang ada pada suatu ruas jalan. Pendeteksi kendaraan tersebut dilakukan oleh rekaman CCTV pada lampu lalu lintas yang datanya diambil dan data berupa citra gambar akan di proses oleh matlab. Blok diagram aplikasi penghitung kendaraan seperti pada gambar 3.1.



Gambar 3. 1 Blok Diagram Aplikasi Penghitung Kendaraan

Gambar blok diagram diatas, memperoleh dari visual yang sudah diambil dari CCTV akan diproses oleh *image processing* pada matlab menggunakan metode YOLO *object detection* untuk menghitung berapa banyak kendaraan yang berada pada salah satu ruas jalan. Program YOLO *Detection Object* hanya akan menghitung kendaraan jika kendaraan melewati garis berwarna kuning yang berfungsi menghitung kendaraan yang lewat seperti pada Gambar 3.2.

UNIVERSITAS Dinamika

Gambar 3. 2 Garis Pendeteksi Kendaraan yang Lewat

3.2 Perancangan Program

Penghitung kendaraan dirancang pada berbagai kondisi lingkungan dengan cahaya dan status lalu lintas berubah. Dalam sistem yang digunakan, sistem menerima lalu lintas dari video kemudian memberikan kotak penghitung pada jalan raya untuk deteksi benda bergerak melewatinya. Sistem akan dijelaskan sebagai berikut.

3.2.1 Sistem Pre-Processing

Tahapan saat proses mengklasifikasi objek gambar secara manual menggunakan *software*. User dapat mengklasifikasi semua objek dan menyimpan hasil gambar beserta *.txt* ke folder data.

3.2.2 Learning

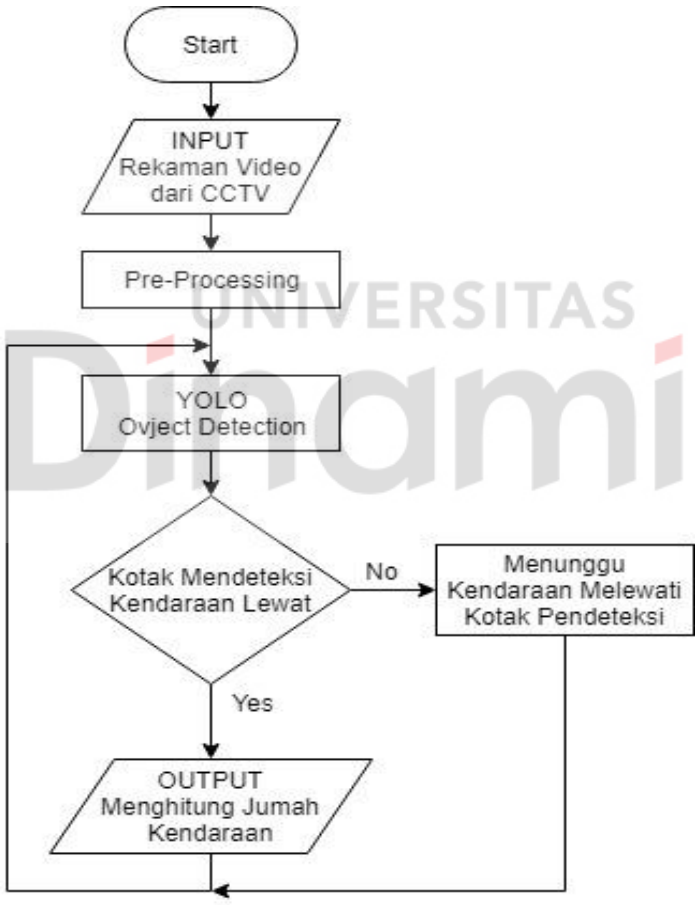
Tahap saat user mengandakan darknet setelah menginstall repo. Kemudian *training* kembali menggunakan model gambar yang sudah ada. Setelah proses *training* selesai maka akan mendapatkan file model yang akan digunakan untuk mendeteksi objek.

3.2.3 Pendeteksi

Jika aplikasi mendeteksi adanya kendaraan yang sedang lewat pada kotak maka program akan menambahkan *counter* +1 yang berfungsi untuk menghitung kendaraan yang lewat pada satu duarasi video.

3.3 Flowchart Sistem Program Software

Untuk mempermudah pembuatan program *software* diperlukan beberapa tahapan yaitu dengan membuat *flowchart* sistem kerja program terlihat sebagai berikut.



Gambar 3. 3 Flowchart Program YOLO Object Detection

Berikut adalah penjelasan dari Gambar 3.3 Flowchart Program YOLO Object Detection:

1. Input Rekaman Video ialah ketika memasukan rekaman video yang dibutukan program untuk mendeteksi kendaraan yang lewat.

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

UNIVERSITAS Dinamika

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini membahas tentang banyaknya hasil dari pengujian dan hasil penelitian Tugas Akhir ini. Dengan tujuan untuk mengetahui tingkat keberhasilan dari perancangan sistem yang telah diajukan dan dikerjakan. Pengujian dilakukan meliputi uji aplikasi menggunakan Metode YOLO, uji deteksi kendaraan beroda 4 dan lebih, dan penghitung kendaraan yang terdeteksi melintas.

4.1 Uji Aplikasi Menggunakan Metode YOLO

4.1.1 Tujuan Pengujian Menggunakan Metode YOLO

Untuk bisa menguji aplikasi penghitung kendaraan maka diperlukan metode yolo untuk dapat mendeteksi kendaraan menggunakan image processing.

UNIVERSITAS Dinamika

4.1.2 Aplikasi Digunakan Untuk Pengujian Metode YOLO

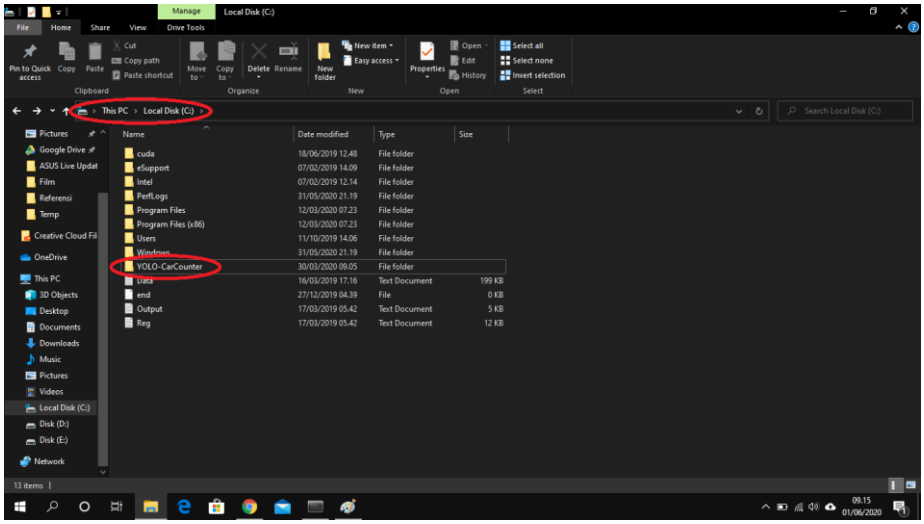
Untuk menguji program YOLO maka diperlukan beberapa aplikasi sebagai berikut:

1. Anaconda
2. Library CUDA Toolkit
3. Library CuDNN
4. Aplikasi Metode YOLO
5. Weight File
6. Video

4.1.3 Prosedur Pengujian Program YOLO

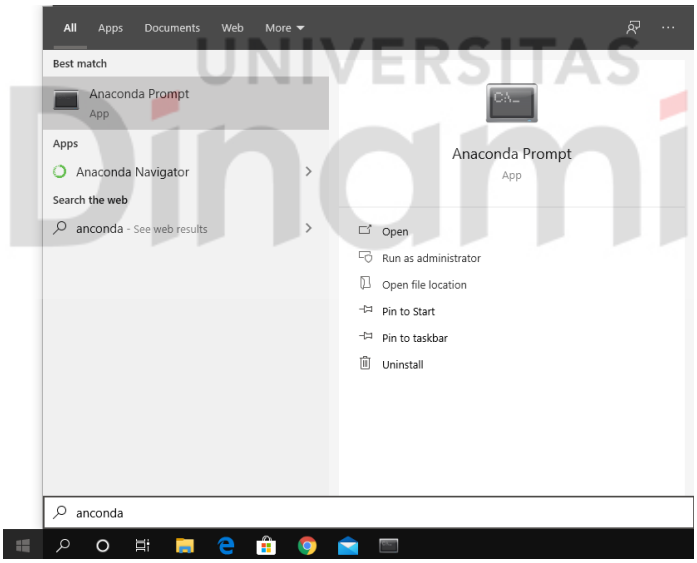
Langkah-langkah yang dilakukan untuk pengujian ini adalah sebagai berikut:

1. Meng-copy file YOLO yang akan dipakai pada Local Disk C untuk mempermudah akses.



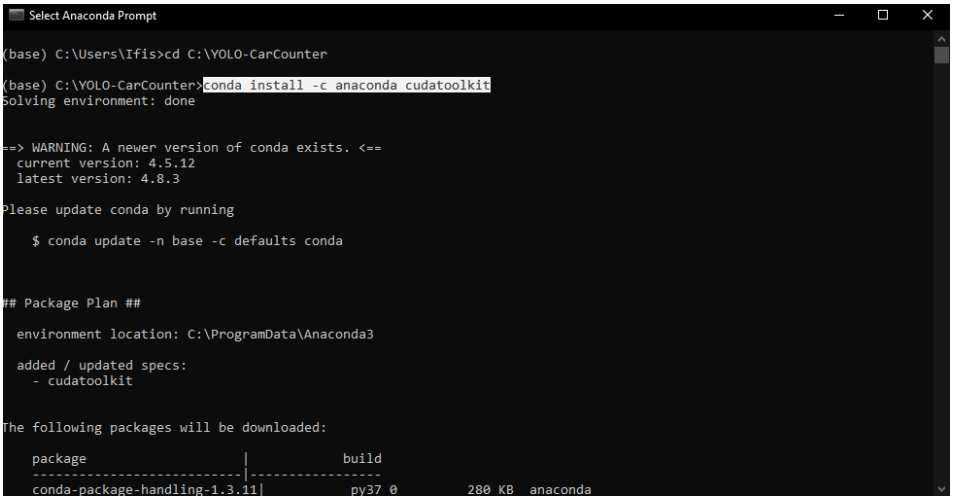
Gambar 4. 1 Tempat File YOLO

2. Memastikan di Laptop atau PC sudah terinstall *software* python Anaconda terbaru dan membuka jendela Anaconda Prompt.



Gambar 4. 2 Buka Anaconda Prompt

3. Setelah jendela Anaconda Prompt terbuka *install library* cudatoolkit didalam file YOLO.

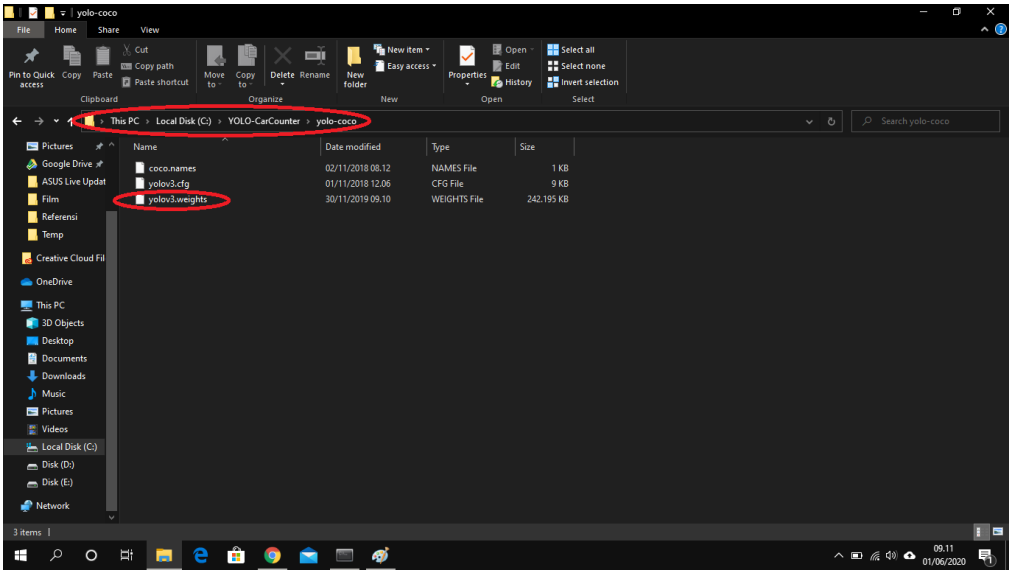


Gambar 4. 3 Instal cudatoolkit

4. Dan setelah cudatoolkit selesai terinstal saatnya *install library* cudnn

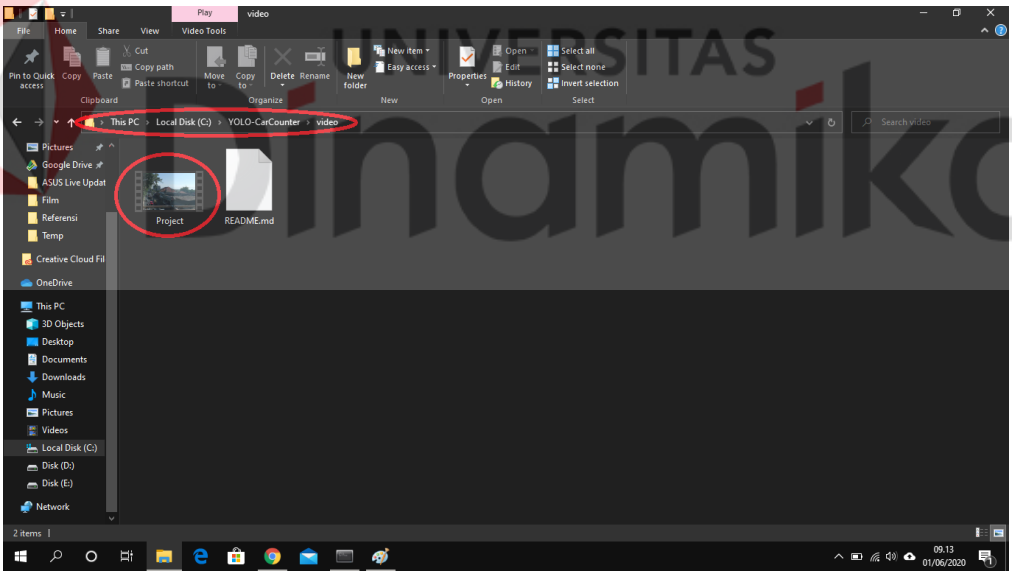
Gambar 4. 4 Instal cudnn

5. Setelah itu mencari weight file yang cocok dengan program YOLO yang dipakai karena pada program ini menggunakan YOLO3 maka harus menggunakan weight file YOLO3 dan copy pada file yolo di dalam folder yolo-coco.



Gambar 4. 5 Penempatan Weight File

6. Setelah itu memindah file video yang akan digunakan untuk menjalankan program mendeteksi kendaraan.



Gambar 4. 6 File Video Yang Akan Digunakan

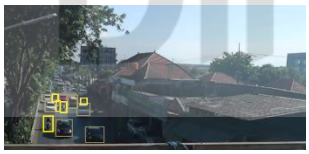
4.1.4 Hasil Pengujian Aplikasi Menggunakan Metode YOLO

Pengujian dilakukan untuk melakukan cek terhadap metode yang dipakai oleh sebuah aplikasi.

Gambar 4. 7 Hasil Pendeteksi Menggunakan Metode YOLO

Selanjutnya adalah dengan pengambilan data dengan cara mengamati berapa banyak Kendaraan yang terdeteksi sebagai gambar. Adapun data dari hasil analisa adalah sebagai berikut:

Tabel 4. 1 Pengujian deteksi kendaraan

Nomor	Gambar	Jumlah kendaraan		Persentase
		Sebenarnya	Deteksi	
1		14	12	86%
2		6	5	83%
3		9	8	89%
4		7	7	100,00%

UNIVERSITAS Dinamika

Nomor	Gambar	Jumlah kendaraan		Persentase
		Sebenarnya	Deteksi	
5		13	11	85%
	Rata – rata			89%

4.2 Uji Deteksi Kendaraan Beroda 4 dan Lebih

4.2.1 Tujuan Uji Deteksi Kendaraan Beroda 4 dan Lebih

Pengujian ini dilakukan untuk memastikan hanya kendaraan dengan menggunakan roda 4 dan lebih yang akan terdeteksi oleh aplikasi.

4.2.2 Aplikasi Uji Deteksi Kendaraan Beroda 4 dan Lebih

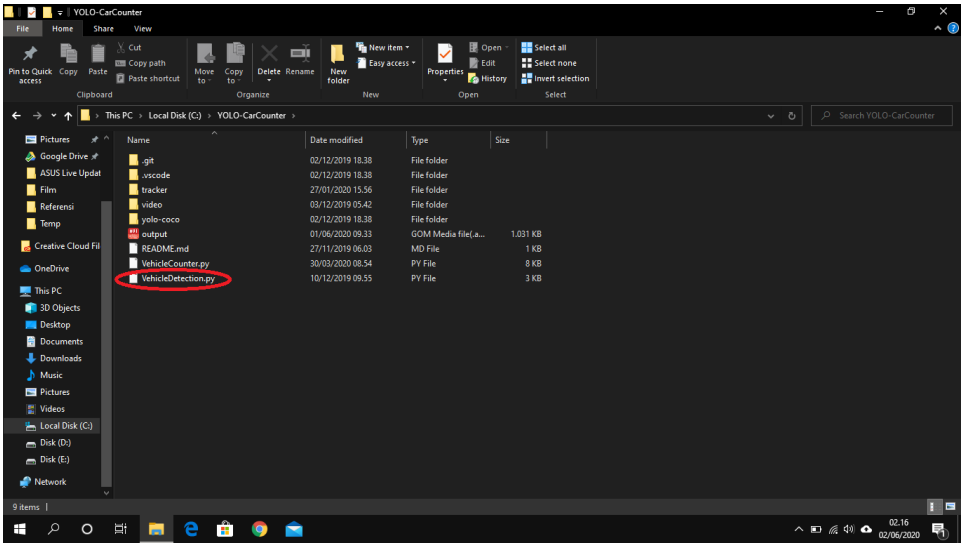
Untuk melakukan percobaan uji deteksi kendaraan maka diperlukan beberapa aplikasi sebagai berikut:

1. Anaconda Prompt
2. Aplikasi Metode YOLO
3. Video

4.2.3 Prosedur Pengujian Deteksi Kendaraan Beroda 4 dan Lebih

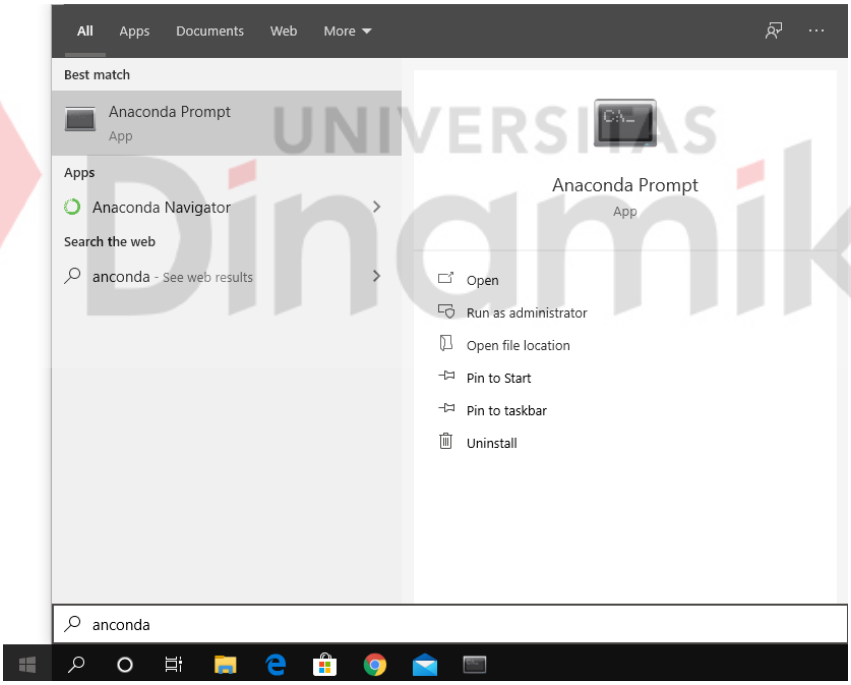
Langkah-langkah yang dilakukan untuk pengujian ini adalah sebagai berikut:

1. Memastikan program Python yang sudah diperbarui sudah ada pada *directory* file YOLO.



Gambar 4. 8 Program Phyton Pendeteksi Kendaraan

2. Membuka jendela Anaconda Prompt pada Laptop atau PC.

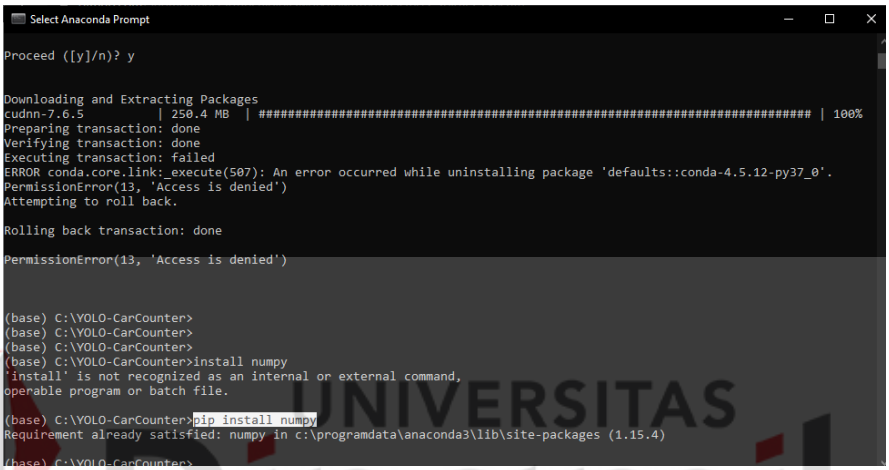


Gambar 4. 9 Buka Anaconda Prompt

3. Jika anaconda Promtp sudah terbuka mengarahkan *directory* ke file program YOLO.

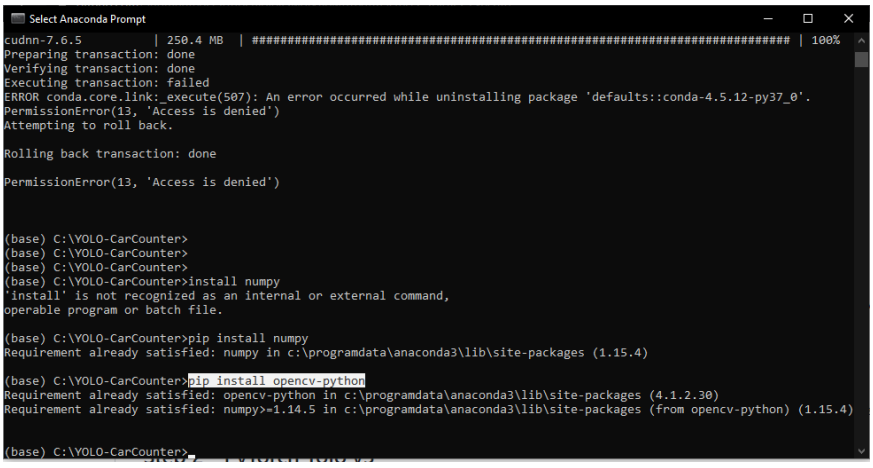
Gambar 4. 10 Anaconda Prompt ke *Directory* File YOLO

4. Setelah itu meng-*install library* numpy.



Gambar 4. 11 Install Library Numpy

5. Setelah itu meng-*install library* opencv-python.



Gambar 4. 12 *Install Library* Python OpenCV

4.2.4 Hasil Pengujian Deteksi Kendaraan Beroda 4 dan Lebih

Dalam pengujian ini sudah terlihat bahwa yang terdeteksi hanya kendaraan yang memiliki roda 4 dan lebih oleh aplikasi menggunakan metode YOLO.

Gambar 4. 13 Hasil Pendeteksi Kendaraan

Pada pengujian kali ini program YOLO sudah bisa membedakan kendaraan dengan unsur yang lain. Ditandai dengan adanya kotak berwarna hijau yang ada disebuah kendaraan. Dengan pendeteskian kali ini akan lebih mudah untuk dapat menghitung kendaraan yang lewat pada jalan raya di video.

Selanjutnya adalah dengan pengambilan data dengan cara mengamati berapa banyak Kendaraan dengan roda 4 atau lebih yang terdeteksi sebagai gambar. Adapun data dari hasil analisa adalah sebagai berikut:

Tabel 4. 2 Pengujian deteksi kendaraan beroda 4 dan lebih

Nomor	Gambar	Jumlah kendaraan		Persentase
		Sebenarnya	Deteksi	
1		7	7	100%
2		6	5	83%

Nomor	Gambar	Jumlah kendaraan		Persentase
		Sebenarnya	Deteksi	
3		7	6	86%
4		5	5	100%
5		8	7	88%
	Rata – rata			91%

4.3 Pengujian Program Penghitung Kendaraan

4.3.1 Tujuan Pengujian Program Penghitung Kendaraan

Pengujian ini dilakukan untuk mempermudah manusia dalam menghitung kendaraan-kendaraan yang lewat pada jalan raya.

4.3.2 Aplikasi digunakan Pada Pengujian Penghitung Kendaraan

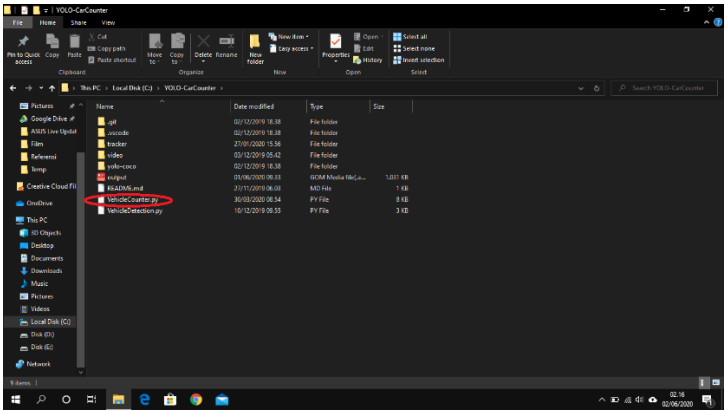
Untuk melakukan percobaan uji kendaraan maka diperlukan beberapa aplikasi sebagai berikut:

1. Anaconda Prompt
2. Aplikasi Metode YOLO
3. Video
4. *Library* Scipy

4.3.3 Prosedur Pengujian Penghitung Kendaraan Melintas

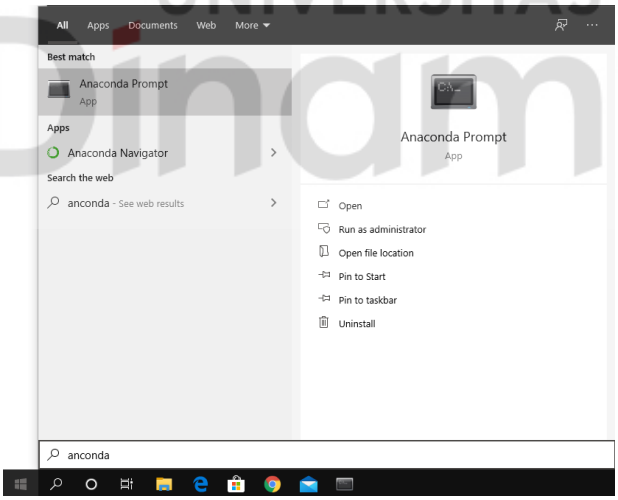
Langkah-langkah yang dilakukan untuk pengujian ini adalah sebagai berikut:

1. Memastikan program Python yang sudah diperbarui sudah ada pada *directory* file YOLO.



Gambar 4. 14 Program Phyton Pendeteksi Kendaraan

2. Membuka jendela Anaconda Prompt pada Laptop atau PC.

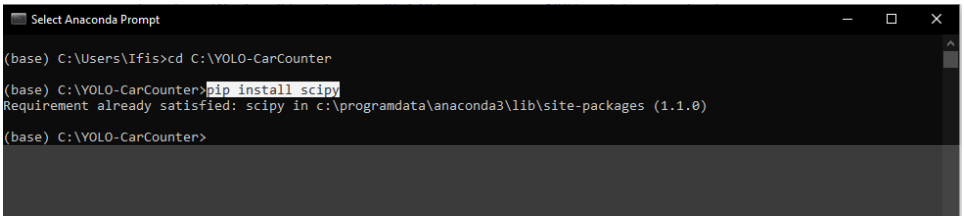


Gambar 4. 15 Buka Anaconda Prompt

3. Jika anaconda Promtp sudah terbuka mengarahkan *directory* ke file program yolo.

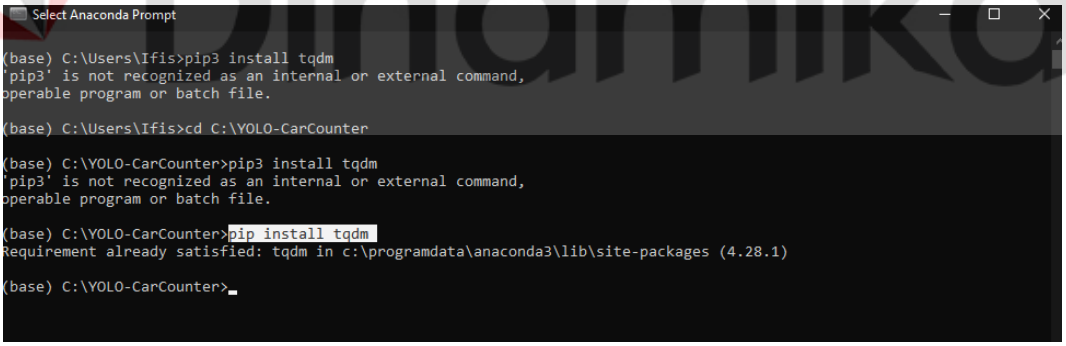
Gambar 4. 16 Anaconda Prompt ke *Directory* File YOLO

4. Setelah itu meng-install library scipy.



Gambar 4. 17 Install library scipy

5. Setelah itu mneg-install library tqdm



Gambar 4. 18 Install library tqdm

4.3.4 Hasil Pengujian Penghitung Kendaraan Melintas

Dalam pengujian ini sudah terlihat bahwa kendaraan yang melintas sudah bisa dihitung oleh counter pada aplikasi menggunakan metode YOLO.

Gambar 4. 19 Hasil Hitung Kendaraan Melintas

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

Selanjutnya adalah dengan pengambilan data dengan cara mengamati berapa banyak mobil yang lewat setiap 10 detik sampai pada menit ke 2 dan melihat kecocokan antara counter aplikasi dengan hitungan manual. Proses pengambilan data dilakukan menggunakan 3 sample video yang berdeda dari berbagai sumber yang ada pada internet serta rekaman penulis. Adapun data dari hasil analisa adalah sebagai berikut :

Tabel 4. 3 Hasil pengujian dengan ketelitian aplikasi 60% pada video 1

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	23	11	12
2	ke-20	38	20	18
3	ke-30	44	27	17
4	ke-40	59	35	24
5	ke-50	71	40	31
6	ke-60	83	45	38
7	ke-70	96	54	42
8	ke-80	108	61	47
9	ke-90	117	68	49
10	ke-100	130	77	53
11	ke-110	146	83	63
12	ke-120	160	91	69

Pada Tabel 4.3 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{69}{91}$ x 100%)

= 100% - 76% = 24%

Tabel 4. 4 Hasil pengujian dengan ketelitian aplikasi 70% pada video 1

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	15	11	4
2	ke-20	28	20	8
3	ke-30	35	27	8
4	ke-40	47	35	12
5	ke-50	60	40	20
6	ke-60	68	45	23
7	ke-70	82	54	28
8	ke-80	95	61	34
9	ke-90	104	68	36
10	ke-100	116	77	39
11	ke-110	129	83	46
12	ke-120	141	91	50

Pada Tabel 4.4 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{50}{91}$ x 100%)

= 100% - 55% = 45%

Tabel 4. 5 Hasil pengujian dengan ketelitian aplikasi 80% pada video 1

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	15	11	4
2	ke-20	25	20	5
3	ke-30	34	27	7
4	ke-40	47	35	12
5	ke-50	56	40	16
6	ke-60	66	45	21
7	ke-70	77	54	23
8	ke-80	86	61	25
9	ke-90	94	68	26
10	ke-100	110	77	33
11	ke-110	121	83	38
12	ke-120	131	91	40

Pada Tabel 4.5 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{40}{91}$ x 100%)

= 100% - 44% = 56%

Tabel 4. 6 Hasil pengujian dengan ketelitian aplikasi 60% pada video 2

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	26	11	15
2	ke-20	42	23	19
3	ke-30	62	32	30
4	ke-40	82	42	40
5	ke-50	102	53	49
6	ke-60	115	60	55
7	ke-70	124	66	58
8	ke-80	143	77	66
9	ke-90	163	89	74
10	ke-100	184	100	84
11	ke-110	196	106	90
12	ke-120	211	116	95

Pada Tabel 4.6 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{95}{116}$ x 100%)

= 100% - 82% = 18%

Tabel 4 7 Hasil pengujian dengan ketelitian aplikasi 70% pada video 2

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	21	11	10
2	ke-20	37	23	14
3	ke-30	57	32	25
4	ke-40	77	42	35
5	ke-50	96	53	43
6	ke-60	107	60	47
7	ke-70	116	66	50
8	ke-80	134	77	57
9	ke-90	151	89	62
10	ke-100	172	100	72
11	ke-110	183	106	77
12	ke-120	196	116	80

Pada Tabel 4.7 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{80}{116}$ x 100%)

= 100% - 69% = 31%

Tabel 4. 8 Hasil pengujian dengan ketelitian aplikasi 80% pada video 2

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	20	11	9
2	ke-20	35	23	12

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
3	ke-30	51	32	19
4	ke-40	71	42	29
5	ke-50	92	53	39
6	ke-60	102	60	42
7	ke-70	113	66	47
8	ke-80	127	77	50
9	ke-90	147	89	58
10	ke-100	169	100	69
11	ke-110	180	106	74
12	ke-120	192	116	76

Pada Tabel 4.8 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{76}{116}$ x 100%)

= 100% - 66% = 34%

Tabel 4. 9 Hasil pengujian dengan ketelitian aplikasi 60% pada video 3

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	18	9	9
2	ke-20	47	26	21
3	ke-30	80	40	40
4	ke-40	107	54	53
5	ke-50	137	71	66
6	ke-60	166	88	78
7	ke-70	196	103	93
8	ke-80	225	119	106
9	ke-90	261	135	126
10	ke-100	279	145	134
11	ke-110	302	158	144
12	ke-120	333	175	158

Pada Tabel 4.9 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat

UNIVERSITAS Dinamika

banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{158}{175}$ x 100%)

= 100% - 90% = 10%

Tabel 4. 10 hasil pengujian dengan ketelitian aplikasi 70% pada video 3

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	15	9	6
2	ke-20	48	26	22
3	ke-30	82	40	42
4	ke-40	111	54	57
5	ke-50	141	71	70
6	ke-60	174	88	86
7	ke-70	206	103	103
8	ke-80	220	119	101
9	ke-90	266	135	131
10	ke-100	281	145	136
11	ke-110	304	158	146
12	ke-120	332	175	157

Pada Tabel 4.10 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{157}{175}$ x 100%)

= 100% - 90% = 10%

Tabel 4. 11 Hasil pengujian dengan ketelitian aplikasi 80% pada video 3

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
1	ke-10	12	9	3
2	ke-20	50	26	24
3	ke-30	82	40	42
4	ke-40	111	54	57
5	ke-50	135	71	64

Nomor	Detik	Jumlah Kendaraan		
		Terdeteksi	Manual	Deteksi Error
6	ke-60	162	88	74
7	ke-70	192	103	89
8	ke-80	215	119	96
9	ke-90	245	135	110
10	ke-100	264	145	119
11	ke-110	284	158	126
12	ke-120	310	175	135

Pada Tabel 4.11 pengambilan data dengan menguji keakuratan penghitungan deteksi aplikasi dan penghitungan manual dapat disimpulkan bahwa masih terdapat banyak deteksi error. Karena data keberhasilan sudah ditemukan maka dapat melakukan proses perhitungan keakuratan deteksi aplikasi dengan cara.

Keakuratan Data = 100% - (deteksi error / perhitungan manual x 100%)

= 100% - ($\frac{135}{175}$ x 100%)

= 100% - 77% = 23%

UNIVERSITAS Dinamika

BAB V
PENUTUP

5.1 Kesimpulan

Dari banyaknya hasil pembahasan di atas dapat disimpulkan beberapa dianataranya adalah :

1. Sistem aplikasi penghitung kendaraan yang melintas di jalan raya menggunakan bantuan dari metode YOLO *object detection* yang dimodifikasi sehingga tidak hanya mendeteksi kendaraan tapi juga dapat menghitung kendaraan yang melewatinya.
2. Sistem aplikasi penghitung kendaraan yang melintas tidak hanya menggunakan metode YOLO *object detection* melainkan juga menggunakan beberapa *libray* yang ada pada aplikasi Python Anaconda.
3. Hasil dari pengujian aplikasi dengan metode YOLO object detection sudah bisa membedakan kendaraan dengan roda 4 atau lebih ditandai dengan deteksi kotak berwarna hijau pada kendaraan yang berada di frame video.
4. Hasil pengujian aplikasi penghitung kendaraan berdasarkan metode YOLO object detection telah berhasil menghitung jumlah kendaran yang melewati sensor deteksi walaupun dengan nilai keakuratan yang belum mencapai 60%.

5.2 Saran

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

1. Menambahkan semua jenis kendaraan yang terdeteksi tidak hanya mendeteksi kendaraan dengan roda 4 atau lebih.
2. Dapat menambahkan nilai akurasi dari aplikasi penghitung kendaraan berdasarkan metode YOLO *object detection* yang telah dibuat penulis.
3. Membandingkan nilai akurasi dari aplikasi penghitung kendaraan berdasarkan metode YOLO *object detection* dengan metode-metode lainnya.

4. Dapat digunakan disaat intensitas cahaya rendah dengan bantuan rekaman kamera yang mempuni.

UNIVERSITAS Dinamika

DAFTAR PUSTAKA

Agrawal, V. (2018). *Melatih Model Klasifikasi Gambar Dengan Membuat Machine Learning.* Jakarta: https://code.tutsplus.com/id/articles/training-an-image-classification-model-with-create-ml--cms-31617.

Antares, A. (2019). *PENGENALAN FDEEP LEARNING MENGGUNAKAN Convolutional Neural Network (CNN).* Surabaya: https://medium.com/@arum.antares9/pengenalan-fdeep-learning-menggunakan-convolutional-neural-network-cnn-67d9b5f9556a.

Dadang, W. (2018). *Memahami Kecerdasan Buatan berupa Deep Learning dan Machine Learning.* Palembang: https://warstek.com/2018/02/06/deepmachinelearning/.

Hidayati, Q. (2017). *Kendali Lampu Lalu Lintas dengan Deteksi Kendaraan.* Jogjakarta: JNTETI, Vol. 6, No. 2.

Karlina, O. E., & Indarti, D. (2019). *Pengenalan Objek Makanan Cepat Saji Pada Video Dan Real Time Webcam Menggunakan Metode You Look Only Once (Yolo).* Jakarta: Jurnal Ilmiah Informatika Komputer, Vol 24, No 3.

MC.AI. (2018). *Deep Learning : Klasifikasi Image dengan Convolutional Neural Network (CNN).* Jakarta: https://mc.ai/deep-learning-klasifikasi-image-dengan-convolutional-neural-network-cnn-menggunakan-keras-di/.

R., A. Y. (2018). *Fully-Connected Layer CNN dan Implementasinya.* Yogyakarta: https://machinelearning.mipa.ugm.ac.id/2018/06/25/fully-connected-layer-cnn-dan-implementasinya/.

Sena, S. (2017). *Pengenalan Deep Learning Part 7 : Convolutional Neural Network (CNN).* Surabaya: https://medium.com/@samuelsena/pengenalan-deep-learning-part-7-convolutional-neural-network-cnn-b003b477dc94.

Syaikhoni, A., & Ariyadi, A. (2018). *Deteksi Objek Dengan Tensorflow Object Detection API.* Jakarta: https://mti.binus.ac.id/2018/12/26/deteksi-objek-dengan-tensorflow-object-detection-api/.

UNIVERSITAS Dinamika