TUGAS AKHIR –TM184835

SIMULASI AUTONOMOUS ROBOT MENGGUNAKAN NVIDIA JETBOT DALAM KLASIFIKASI LAMPU LALU LINTAS.

BONIFASIUS JEREMY BASKORO

NRP 02111640000104

Dosen Pembimbing

Alief Wikarta S.T., M.Sc. Eng., Ph.D.

NIP 198202102006041002

Program Studi Teknik Mesin

Departemen Teknik Mesin

Fakultas Teknologi Industri Dan Rekayasa Sistem

Institut Teknologi Sepuluh Nopember

Surabaya

2022

TUGAS AKHIR –TM184835

SIMULASI AUTONOMOUS ROBOT MENGGUNAKAN NVIDIA JETBOT DALAM KLASIFIKASI LAMPU LALU LINTAS.

BONIFASIUS JEREMY BASKORO

NRP 02111640000104

Dosen Pembimbing

Alief Wikarta S.T., M.Sc. Eng., Ph.D.

NIP 198202102006041002

Program Studi Teknik Mesin

Departemen Teknik Mesin

Fakultas Teknologi Industri Dan Rekayasa Sistem

Institut Teknologi Sepuluh Nopember

Surabaya

2022

FINAL PROJECT –TM184835

AUTONOMOUS ROBOT SIMULATION USING NVIDIA JETBOT IN TRAFFIC LIGHT CLASSIFICATION

BONIFASIUS JEREMY BASKORO

NRP 02111640000104

Advisor

Alief Wikarta S.T., M.Sc. Eng., Ph.D.

NIP 198202102006041002

Study Program of Mechanical Engineering

Department of Mechanical Engineering

Faculty of Industrial Technology and Systems Engineering

Institut Teknologi Sepuluh Nopember

Surabaya

2022

LEMBAR PENGESAHAN

SIMULASI AUTONOMOUS ROBOT MENGGUNAKAN NVIDIA JETBOT DALAM KLASIFIKASI LAMPU LALU LINTAS.

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Teknik

Program Studi S-1 Departemen Teknik Mesin

Fakultas Teknologi Industri dan Rekayasa Sistem

Institut Teknologi Sepuluh Nopember

Oleh : **Bonifasius Jeremy Baskoro**

NRP. 02111640000104

Disetujui oleh Tim Penguji Tugas Akhir:

1. Alief Wikarta, S.T., M.Sc. Eng., Ph.D. Pembimbing
2. Mohammad Khoirul Effendi, S.T., M. Sc. Eng., Ph.D. Penguji
3. Ir. Julendra B. Ariatedja, M.T. Penguji
4. Dr.Eng. Yohanes, S.T., M. Sc. Penguji

SURABAYA

Juli, 2022

APPROVAL SHEET

AUTONOMOUS ROBOT SIMULATION USING NVIDIA JETBOT IN TRAFFIC LIGHT CLASSIFICATION

FINAL PROJECT

Submitted to fulfill one of the requirements

for obtaining a degree Bachelor of Engineering at

Undergraduate Study Program of Mechanical Engineering

Department of Mechanical Engineering

Faculty of Industrial Technology and Systems Engineering

Institut Teknologi Sepuluh Nopember

By: **Bonifasius Jeremy Baskoro**

NRP. 02111640000104

Approved by Final Project Examiner Team:

1. Alief Wikarta, ST., M. Sc. Eng., Ph.D. — Advisor
2. Mohammad Khoirul Effendi, S.T., M. Sc. Eng., Ph.D. — Examiner
3. Ir. Julendra B. Ariatedja, M.T. — Examiner
4. Dr.Eng. Yohanes, S.T., M. Sc. — Examiner

SURABAYA

July, 2022

"Halaman ini sengaja dikosongkan"

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP	: Bonifasius Jeremy Baskoro/02111640000104
Departemen	: Teknik Mesin
Dosen Pembimbing / NIP	: Alief Wikarta, ST., MSc. Eng. Ph.D /198202102006041002

dengan ini menyatakan bahwa Tugas Akhir dengan judul "**SIMULASI AUTONOMOUS ROBOT MENGGUNAKAN NVIDIA JETBOT DALAM KLASIFIKASI LAMPU LALU LINTAS**." adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 1 Agustus 2022

Mengetahui

Dosen Pembimbing,

Alief Wikarta, S.T., MSc. Eng., Ph.D.

NIP. 198202102006041002

Mahasiswa,

Bonifasius Jeremy Baskoro

NRP.02111640000104

“Halaman ini sengaja dikosongkan”

STATEMENT OF ORIGINALITY

The undersigned below:

Nama of student / NRP	: Bonifasius Jeremy Baskoro/02111640000104
Department	: Mechanical Engineering
Advisor / NIP	: Alief Wikarta, S.T., MSc. Eng. Ph.D /198202102006041002

hereby declare that the Final Project with the title of "**AUTONOMOUS ROBOT SIMULATION USING NVIDIA JETBOT IN TRAFFIC LIGHT CLASSIFICATION.**" is the result of my own work, is original, and is written by following the rules of scientific writing.

If in the future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with the provisions that apply at Institut Teknologi Sepuluh Nopember.

Surabaya, 1st August 2022

Acknowledge

Advisor,

Alief Wikarta, S.T., M. Sc. Eng., Ph.D.
NIP. 198202102006041002

Student,

Bonifasius Jeremy Baskoro
NRP.02111640000104

“Halaman ini sengaja dikosongkan”

SIMULASI AUTONOMOUS ROBOT MENGGUNAKAN NVIDIA JETBOT DALAM KLASIFIKASI LAMPU LALU LINTAS.

Nama Mahasiswa / NRP : Bonifasius Jeremy Baskoro / 02111640000181

Departemen : Teknik Mesin FTIRS-ITS

Dosen Pembimbing : Alief Wikarta, ST., MSc. Eng. Ph.D

ABSTRAK

Penggunaan *autonomous robot* untuk membantu pekerjaan manusia sudah mulai dilakukan. Mulai dari industri manufaktur, industri logistic, maupun industri medis. Tujuan dari penggunaan *autonomous robot* adalah untuk memudahkan manusia serta melakukan kegiatan pekerjaan lebih efektif. *Autonomous robot* dilengkapi dengan kamera untuk mengetahui lingkungan sekitarnya. Masukan dari kamera diproses oleh komputer menggunakan beberapa metode seperti *computer vision, machine learning, deep learning.* Salah satu alat yang dapat digunakan untuk menerapkan *autonomous robot* adalah NVIDIA Jetbot. Dengan tujuan agar *autonomous robot* dapat berinteraksi dengan baik dengan lampu lalu lintas maka dilakukan pengaplikasian menggunakan NVIDIA Jetbot.

Salah satu metode untuk NVIDIA Jetbot dapat berfungsi sebagai *autonomous robot delivery* adalah dengan dapat mengklasifikasi lampu lalu lintas seperti skenario umumnya pada jalanan di Indonesia. Pada penelitian ini digunakan ResNet karena arsitektur ResNet dinilai mempunyai nilai akurasi yang tinggi serta dengan model yang mempunyai parameter yang sedikit. NVIDIA Jetbot akan menjalankan fungsi klasifikasi lampu lalu lintas dengan model ResNet dan input dari kamera Jetbot. Pengujian klasifikasi lampu lalu lintas dilakukan pada 4 tingkat kecepatan yang berbeda, 2 variasi kondisi pencahayaan lintasan, dan serta dataset *training* yang berbeda yaitu terdiri dari 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi dan tanpa augmentasi, serta dataset berupa gambar diambil dari kamera Jetbot dengan jumlah 250 dan 500 gambar.

Hasil Performa Jetbot dalam klasifikasi lampu lalu lintas pada pengujian di kondisi pencahayaan terang dengan kecepatan berturut-turut sebesar 0,3, 0,4, 0,6, dan 0,8 untuk data *training* 500 gambar Bosch Small Traffic Light Dataset tanpa augmentasi yaitu 50,67%, 40,67%, 34,67%, 29,33%. Untuk data training 500 Gambar Bosch Small Traffic Light Dataset dengan augmentasi yaitu 58%, 55,33%, 44,67%, dan 30,67%. Untuk data *training* 250 gambar diambil dari kamera Jetbot yaitu 93,33%, 92%, 82,67%, dan 70,67%. Untuk data training 500 gambar diambil dari kamera Jetbot yaitu 96%, 94,67%, 86,67%, dan 72,67%. Pada kondisi pencahayaan gelap dengan kecepatan berturut-berturut sebesar sebesar 0,3, 0,4, 0,6, dan 0,8 untuk data *training* 500 gambar Bosch Small Traffic Light Dataset tanpa augmentasi yaitu 40,67%, 39,33%, 32,67%, dan 28,00%. Untuk data training 500 Gambar Bosch Small Traffic Light Dataset dengan augmentasi yaitu 59,33%, 56,67%, 53,33%, dan 29,33%. Untuk data *training* 250 gambar diambil dari kamera Jetbot yaitu 93,33%, 94%, 88,67%, dan 68%. Untuk data training 500 gambar diambil dari kamera Jetbot yaitu 97,33%, 94%, 84,67%, dan 82%.

Kata Kunci: *Autonomous Robot, Machine Learning*, *Deep Learning, NVIDIA Jetbot.*

“Halaman ini sengaja dikosongkan”

AUTONOMOUS ROBOT SIMULATION USING NVIDIA JETBOT IN TRAFFIC LIGHT CLASSIFICATION.

Student Name / NRP : **Bonifasius Jeremy Baskoro**

Department : **Mechanical Engineering FTIRS-ITS**

Advisor : **Alief Wikarta, ST., MSc. Eng. Ph.D**

ABSTRACT

The use of autonomous robots to help human work has begun. Starting from the manufacturing industry, the logistics industry, and the medical industry. The purpose of using autonomous robots is to make it easier for humans to carry out work activities more effectively. Autonomous robot is equipped with a camera to know the surrounding environment. Input from the camera is processed by a computer using several methods such as computer vision, machine learning, deep learning. One of the tools that can be used to implement autonomous robots is NVIDIA Jetbot. With the aim that autonomous robots can interact well with traffic lights, an application is made using NVIDIA Jetbot.

One of the methods for the NVIDIA Jetbot to function as an autonomous robot delivery is to be able to classify traffic lights as in general scenarios on roads in Indonesia. In this study, ResNet is used because the ResNet architecture is considered to have a high accuracy value and with a model that has few parameters. NVIDIA Jetbot will perform the traffic light classification function with the ResNet model and input from the Jetbot camera. Traffic light classification tests were carried out at 4 different speed levels, 2 variations of track lighting conditions, and a different training dataset consisting of 500 images of the Bosch Small Traffic Light Dataset with and without augmentation, as well as a dataset in the form of images taken from a Jetbot camera with the number of 250 and 500 images.

Jetbot performance results in traffic light classification on tests in bright lighting conditions with successive speeds of 0.3, 0.4, 0.6, and 0.8 for training data of 500 images of Bosch Small Traffic Light Dataset without augmentation, which is 50 ,67%, 40.67%, 34.67%, 29.33%. For training data 500 Images of Bosch Small Traffic Light Dataset with augmentation are 58%, 55.33%, 44.67%, and 30.67%. For training data, 250 images were taken from the Jetbot camera, namely 93.33%, 92%, 82.67%, and 70.67%. For training data 500 images were taken from the Jetbot camera, namely 96%, 94.67%, 86.67%, and 72.67%. In dark lighting conditions with successive speeds of 0.3, 0.4, 0.6, and 0.8 for training data of 500 images of Bosch Small Traffic Light Dataset without augmentation, namely 40.67%, 39.33%, 32.67%, and 28.00%. For training data 500 Images of Bosch Small Traffic Light Dataset with augmentation are 59.33%, 56.67%, 53.33%, and 29.33%. For training data, 250 images were taken from the Jetbot camera, namely 93.33%, 94%, 88.67%, and 68%. For training data 500 images were taken from the Jetbot camera, namely 97.33%, 94%, 84.67%, and 82%.

Keywords: *Autonomous Robot, Machine Learning*, *Deep Learning, NVIDIA Jetbot.*

“Halaman ini sengaja dikosongkan”

KATA PENGANTAR

Puji syukur penulis panjatkan ke Tuhan yang Maha Esa, atas rahmat, berkah, dan berkat-Nya sehingga penulis dapat menyelesaikan penyusunan Laporan Tugas Akhir ini.

Penyusunan Tugas Akhir ini selain merupakan salah satu persyaratan yang harus dipenuhi untuk menyelesaikan pendidikan Tingkat Sarjana pada Fakultas Teknik Jurusan Mesin Institut Teknologi Sepuluh Nopember juga dimaksudkan untuk menambah wawasan di bidang *artificial intelligence* untuk *autonomous robot* yang dapat membantu kehidupan manusia. Pada kesempatan ini ijinkan penulis untuk mengucapkan terima kasih dan rasa hormat atas segala bantuan yang telah diberikan kepada penulis sehingga dapat menyelesaikan Laporan Tugas Akhir ini, yaitu kepada:

1. Bapak Alief Wikarta, S.T., M.Sc. Eng. Ph.D. selaku Dosen Pembimbing tugas akhir saya yang telah membimbing, memberi masukan sehingga penulis dapat lebih menyempurnakan Laporan Tugas Akhir ini.
2. Bapak Mohammad Khoirul Effendi, S.T., M.Sc.Eng., Ph.D., Bapak Ir. Julendra B. Ariatedja, M.T., Bapak Dr.Eng. Yohanes, S.T., M.Sc. selaku Dosen Penguji tugas akhir saya yang telah membimbing, memberi masukan sehingga penulis dapat lebih menyempurnakan Laporan Tugas Akhir ini.
3. Seluruh dosen, staf, dan karyawan Jurusan Teknik Mesin FTI-RS ITS atas jasa-jasanya selama penulis menuntut ilmu.
4. Mama, Bapak, Inka, dan Keluarga Soekowati atas dukungan dan doanya.
5. Ubi, Kiming, Yusuf, Afli, Thaariq, Stelon, dan kawan-kawan M59 yang bertemu di TKD atas *support* untuk mendukung dalam penanaman spike.
6. Mat Kafi, Mas Muchtar, Jay, Aziz, Vicky, dan kawan-kawan laboratorium Mekanika Benda Padat dalam dukungannya setiap malam dan berjuang bersama untuk mewujudkan laporan tugas akhir ini.
7. Atwin, Tio, Aliya, Sekar, Dani, dan kawan-kawan ABG Gemas atas dukungannya dalam menyehatkan mental dalam penyusunan Laporan Tugas Akhir ini.
8. Semua pihak yang telah banyak memberikan bantuan yang tidak dapat penulis sebutkan satu persatu sehingga mengantarkan penulis untuk menyelesaikan Laporan Tugas Akhir ini.

Dalam penyusunan Laporan ini tentunya masih banyak terdapat kekurangan, kesalahan dan kekhilafan karena keterbatasan kemampuan penulis, untuk itu sebelumnya penulis mohon maaf yang sebesar-besarnya. Penulis juga mengharapkan kritik dan saran dari semua pihak demi perbaikan yang bersifat membangun atas laporan ini. Akhirnya dengan segala kerendahan hati penulis mengucapkan terima kasih dan semoga laporan ini dapat bermanfaat bagi penulis maupun kita bersama.

Surabaya, Juli 2022

Penulis

“Halaman ini sengaja dikosongkan”

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

“Halaman ini sengaja dikosongkan”

BAB I
PENDAHULUAN

1.1 Latar Belakang

Lampu lalu lintas merupakan alat yang mengendalikan arus lalu lintas dengan menandakan kapan kendaraan harus berjalan dan berhenti secara bergantian dari berbagai arah. *Autonomous robot* yang merupakan alat yang didesain untuk dapat berinteraksi dengan lingkungan sekitarnya tanpa pengendalian dari manusia dapat membantu kehidupan manusia sehari-hari. *Autonomous robot* sudah mulai diadaptasi pada kehidupan sehari-hari, di Amerika Serikat perusahaan besar seperti Uber, Amazon, dan DHL sudah mengadaptasi *autonomous robot* sebagai barang yang dapat membantu proses bisnis. Seperti Uber dan Amazon yang menggunakan *autonomous robot* untuk melakukan proses pengantaran pesanan tentu akan berinteraksi dengan lalu lintas. Di Indonesia dengan trotoar yang relatif kurang baik kondisinya tentu dalam mengadaptasi *autonomous robot* akan berkaitan langsung dengan jalan dan lalu lintas. *Autonomous robot* yang digunakan sebagai pengantar tentunya berbentuk seperti kendaraan yang dapat berjalan menggunakan roda dan berukuran tidak terlalu besar.

Salah satu cara *autonomous robot* dapat berinteraksi dengan lingkungan sekitarnya adalah dengan teknologi *Computer vision(*CV) merupakan bagian dari kecerdasan buatan yang memberi kemampuan untuk komputer dalam mengambil informasi dari gambar, atau video dan hal visual lainnya. Pengembangan kendaraan swakemudi menggunakan CV pada tahun 1988 oleh Thorpe dan kawan kawan yaitu menggunakan kamera untuk mendeteksi jalur pada jalan dengan melihat garis pada jalan serta mengekstraksi informasi terkait pinggiran jalan. Pada tahun 1988 Pomerleau meneliti tentang ALVINN yang merupakan kendaraan swakemudi berfungsi untuk mengikuti jalan menggunakan *neural network* dalam memproses gambar dari kamera yang merupakan gabungan dari CV dan *machine learning*. Pada penelitian Pomerleau mobil NAVLAB berhasil mengemudi sesuai dengan jalur sepanjang 400 meter dalam kecepatan 0.5 meter per sekon.

Penelitian CV khususnya dalam mendeteksi objek sudah berkembang banyak dengan banyaknya algoritma yang dipublikasi. Pendeteksi objek berfungsi sebagai pemberi informasi ke mobil swakemudi tentang lingkungan sekitarnya dengan menggunakan gambar yang diambil oleh kamera. Algoritma klasifikasi gambar yang sudah sering digunakan seperti ResNet, AlexNet, MobileNet, dapat membantu *autonomous robot* dalam mendeteksi objek di sekitar. ResNet merupakan penemuan yang membantu dalam mengembangkan performa *artificial neural network* agar dapat mengantisipasi degradasi performa yang diakibatkan oleh *layer-layer* pada *neural network* menggunakan teknik *skip*, ResNet ditemukan oleh Kaiming He dan kawan-kawan. D*eep convolutional network* untuk mendeteksi fitur dan objek pada gambar input dan fitur terakhir dari layer *deep convolutional network* dimasukan ke ROI *pooling layer* untuk mengambil fitur dan memberi kotak pembatas pada objek yang terdeteksi.

Pada tahun 2020 Kawakura dan Shibasaki menggunakan NVIDIA Jetbot untuk menguji Jetbot dapat mendeteksi objek serta menghindari rintangan. Penelitian ini menggunakan algoritma objek deteksi yaitu YOLO yang berdasarkan *convolutional neural network* dan ResNet. Pengujian ini dilakukan dengan membuat Jetbot berjalan dari titik awal ke titik tujuan dimana ada objek penghalang dan dihitung keberhasilan dari Jetbot untuk berpindah dari titik awal ke titik tujuan. Hasil dari pengujian didapatkan tingkat keberhasilan 56% dan 60% dari titik awal dan titik tujuan dengan 2 variasi.

Tujuan dari penelitian ini adalah menggunakan Jetbot sebagai miniatur robot yang dapat mengenali lampu lintas dan mengklasifikasi warna lampu itu serta memberi respon yang tepat pada jalanan(lintasan). Serta mengetahui bagaimana performa model *machine learning* dalam mendeteksi lampu lalu lintas menggunakan komputer yang tergolong berukuran kecil yaitu NVIDIA Jetson Nano. Jetbot yang digunakan adalah Jetbot Waveshare kit yang menggunakan kamera dan komputer NVIDIA Jetson Nano. Jetbot juga akan diuji dalam 4 variasi kecepatan yaitu 0.3, 0.4, 0.6, dan 0.8 pada tingkatan kecepatan Jetbot dan variasi pencahayaan pada lintasan.

1.2 Rumusan Masalah

Dalam penyusunan tugas akhir ini, dirumuskan beberapa permasalahan yaitu sebagai berikut

1. Berapa kecepatan optimal untuk Nvidia Jetbot mengklasifikasi warna lampu lalu lintas dengan tepat menggunakan algoritma *image classification*?
2. Bagaimana pengaruh variasi pencahayaan lintasan gelap dan terang terhadap NVIDIA Jetbot dalam mendeteksi lampu lalu lintas menggunakan algoritma *image classification*?
3. Bagaimana pengaruh data *training* yang digunakan model *deep learning* untuk *image classification* pada NVIDIA Jetbot dalam mengklasifikasi lampu lalu lintas?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah sebagai berikut.

1. Mengetahui kecepatan optimal untuk Nvidia Jetbot dalam mengklasifikasi warna lampu lalu lintas menggunakan algoritma *image classification*.
2. Mengetahui pengaruh pencahayaan lintasan terhadap akurasi Nvidia Jetbot dalam mengklasifikasi lampu lalu lintas menggunakan algoritma *image classification*.
3. Mengetahui pengaruh data *training* yang digunakan model *deep learning* untuk *image classification* pada NVIDIA Jetbot dalam mengklasifikasi lampu lalu lintas

1.4 Batasan Masalah

Adapun batasan masalah yang digunakan pada penelitian ini adalah sebagai berikut.

1. Alat yang digunakan adalah NVIDIA autonomous machine tipe Waveshare JetBot AI Kit
2. Arsitektur kecerdasan buatan yang digunakan adalah algoritma ResNet18 dengan basis Convolutional Neural Network.
3. Pemrograman menggunakan Python dan JupyterLab
4. *Dataset* yang digunakan adalah *Bosch Small Traffic Light Dataset* dan gambar yang diambil oleh penulis.
5. Lampu lalu lintas yang digunakan merupakan lampu lalu lintas miniatur.
6. Lampu lalu lintas miniatur yang digunakan satu buah.
7. Lampu lalu lintar miniatur terdiri dari lampu merah lingkaran, lampu hijau lingkaran, dan lampu kuning lingkaran.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah untuk mengembangkan penggunaan *autonomous robot* pada NVIDIA Jetbot yang dapat dikatakan sebagai robot menggunakan komputer kecil yang dapat digunakan untuk dijadikan robot yang membantu kehidupan sehari-hari. Khususnya untuk menjadi pengantar logistik yang dapat membantu di masa depan.

“Halaman ini sengaja dikosongkan”

BAB II
DASAR TEORI

2.1 Lalu Lintas di Indonesia

Lalu lintas di dalam Undang-undang No 22 tahun 2009 didefinisikan sebagai gerak Kendaraan dan orang di Ruang Lalu Lintas Jalan, sedangkan yang dimaksud dengan Ruang Lalu Lintas Jalan adalah prasarana yang diperuntukkan bagi gerak pindah Kendaraan, orang, dan/atau barang yang berupa Jalan dan fasilitas pendukung. Partisipan lalu lintas di Indonesia dimulai dari kendaraan bermotor seperti mobil, sepeda motor, dan truk. Pada tahun 2021 berdasarkan data Korps Lalu Lintas Polri total kendaraan bermotor mencapai 143.340.128 unit, 60% dari total kendaraan bermotor berada di Pulau Jawa. Tidak hanya kendaraan bermotor namun juga ada partisipan lalu lintas seperti sepeda, becak, gerobak, dan pejalan kaki. Pada Undang-undang No 22 Tahun 2009, komponen pokok lalu lintas terdiri dari jalan, jalur, lajur, lalu lintas dan angkutan jalan, marka jalan, kendaraan, kendaraan bermotor, kendaraan tidak bermotor, kendaraan bermotor umum, kendaraan umum, sepeda motor, mobil penumpang, mobil bus, pengemudi, pejalan kaki, lalu lintas, ruang lalu lintas jalan, angkutan, jaringan lalu lintas, pengguna jalan, perusahaan angkutan umum, berhenti, rambu lalu lintas, kelancaran lalu lintas.

Gambar 2.1 *Partisipan Lalu Lintas*

Banyaknya komponen pada lalu lintas tentu diperlukan aturan yang mengatur agar terjadinya keamanan dan kenyamanan untuk semua pelaku lalu lintas. Fenomena pelanggar lalu lintas yang menyebabkan tidak lancarnya lalu lintas pun kerap terjadi, pelanggaran lalu lintas di Indonesia sering terjadi dan bahkan terdapat korban. Faktor manusia berperan penting dalam pelanggaran lampu lalu lintas menurut Rahardjo, pelanggaran terjadi karena kesengajaan, kurangnya mengerti terhadap aturan yang berlaku atau tidak melihat ketentuan yang berlaku. Selain itu manusia sebagai pengguna jalan sering melakukan hal tak perlu saat berkendara dan kecelakaan lalu lintas juga diakibatkan karena pengendara yang mabuk dan mengantuk.

2.2 Lampu Lalu Lintas

Lampu lalu lintas menurut UU no.22 tahun 2009 adalah lampu yang mengendalikan arus lalu lintas lainnya. Lampu ini yang menandakan kapan kendaraan harus berjalan dan berhenti secara bergantian dari berbagai arah. Pengaturan lalu lintas di persimpangan jalan dimaksudkan untuk mengatur pergerakan kendaraan pada masing-masing kelompok pergerakan kendaraan agar dapat bergerak secara bergantian sehingga tidak saling mengganggu antar-arus yang ada. Penemu lampu lalu lintas pertama adalah Lester Farnsworth Wire. Pada tahun 1860 di London sudah dimulai dengan tanda jalan dan berhenti menggunakan palang bertuliskan "stop" dan "go", dan pada malam hari menggunakan lampu gas yang dioperasikan petugas, usulan ini berasal dari John Peake Knight karena terinspirasi dari lampu penanda kereta pada waktu itu. Lalu pada tahun 1912, Lester Farnsworth Wire seorang perwira polisi di Salt Lake City membuat lampu lalu lintas dengan 4 sisi dan tiang tinggi yang ditaruh di tengah persimpangan untuk menandakan kapan kendaraan harus jalan dan berhenti. Pada 1920 akhirnya ditemukan lampu lalu lintas dengan 3 sinyal, dengan bertambahnya sinyal lampu kuning yang berarti "hati-hati", yang dipatenkan oleh William Potts. Lampu lalu lintas pada zaman ini sudah mempunyai fitur yang banyak, seperti dapat menentukan lamanya durasi tiap sinyal lampu berdasarkan pemakai jalan menggunakan *computer vision*, ada yang terintegrasi secara langsung dengan petugas.

Gambar 2.2 *Lampu lalu lintas penemuan William Potts*

Pada tahun 2019, Kota DKI Jakarta mempunyai total 96 lampu lalu lintas yang mempunyai teknologi Intelligent Transportation System Area Traffic Control System (ITS ATCS). Kegunaan teknologi ATCS adalah untuk mengintegrasikan semua lampu lalu lintas yang dapat membantu melancarkan lalu lintas pada kawan tersebut. Informasi tentang keadaan lalu lintas dapat tersampaikan dengan cepat karena terintegrasi dengan sistem ITS ATCS. Selain itu dapat digunakan sebagai alat untuk menertibkan pelanggar lalu lintas.

2.3 Autonomous Robot

Autonomous Robot adalah robot yang dapat melakukan tindakan tanpa membutuhkan pengendalian dari manusia. *Autonomous Robot* didesain dan direkayasa agar dapat menghadapi lingkungan sekitarnya dan berinteraksi sesuai kegunaanya. Kegunaan *autonomous robot* sudah banyak mulai dari industri manufaktur, industri kesehatan, dan industri logistik. *Autonomous robot* dibuat untuk membantu kehidupan manusia dan membuat pengerjaan hal lebih efektif dan efisien. *Autonomous robot* dapat berinteraksi dengan lingkungan sekitarnya karena menggunakan sensor-sensor, dan juga logika pemrograman yang ditaruh dan diproses oleh komputer. Selain

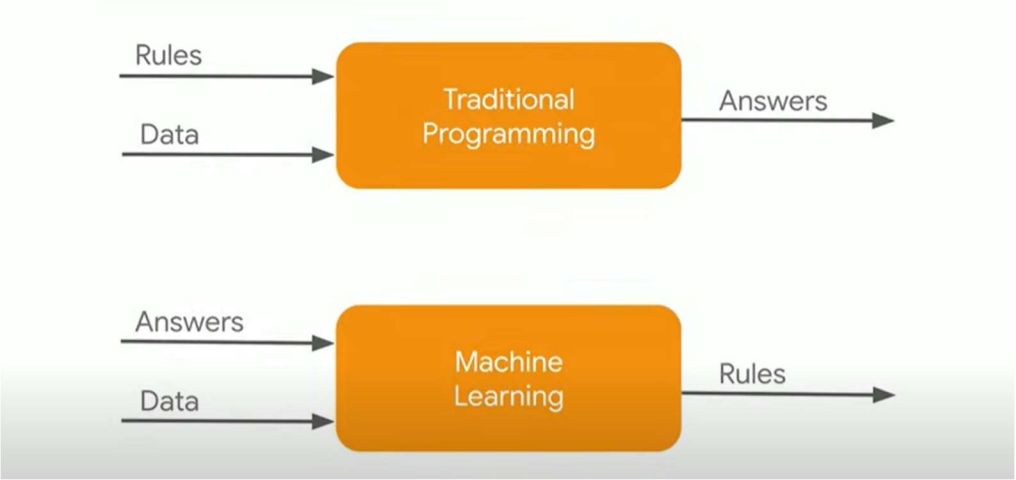
membutuhkan sensor beberapa robot juga membutuhkan aktuator untuk berinteraksi dan mengerjakan tugasnya. Seperti motor listrik yang digunakan untuk menggerakan bagian pada robot. Contohnya untuk robot pembersih rumah sensor yang digunakan adalah sensor ultrasonik, infrared, dan sensor cahaya, serta aktuator yang dipunyai adalah roda dengan motor listrik, serta vakum.

Gambar 2.3 *Robot pengantar makanan*

Pada gambar 2.3 merupakan gambar dari robot pengantar makanan yang dipunyai oleh Uber. Penerapan *autonomous robot* sudah banyak pada kehidupan sehari-hari. Dengan *autonomous robot* manusia dapat terbantu dalam mengerjakan tugasnya sehingga membuat proses yang lebih efisien dan efektif.

2.4 Machine Learning

Penerapan *machine learning* dalam beberapa tahun terakhir menjadi hal yang umum dalam kehidupan sehari-hari. Dari rekomendasi makanan yang akan dipesan, film mana yang akan ditonton, dan coffeshop mana yang akan dikunjungi. Banyak juga situs-situs web yang sudah menggunakan machine learning, seperti Facebook dan Netflix. *Machine Learning* dapat didefinisikan sebagai pemrograman komputer untuk mengembangkan model yang dapat belajar dengan menggunakan data contoh (Alpaydin, 2020). Machine Learning memiliki beberapa metode pembelajaran, yaitu *supervised learning*,*unsupervised learning, reinforcement learning*. Konsep dasar *machine learning* dapat dilihat pada gambar dibawah.



Gambar 2.4 *Perbedaan machine learning dan pemrograman tradisional*

Perbedaan *machine learning* dan pemrograman tradisional adalah letak input yang dimasukkan. Pada pemrograman tradisional masukan dari program merupakan aturan atau logika, serta data yang digunakan untuk mencari jawaban ataupun *output*. Pada konsep *machine learning*, masukan yang digunakan adalah jawaban, dan data. Algoritma *machine learning* akan mencari pola yang terbaik untuk mendapatkan kesinambungan antara data serta *output*-nya akan berbentuk aturan atau logika.

Machine learning dapat didefinisikan sebagaimetode komputasi berdasarkan pengalaman untuk meningkatkan performa ataumembuat prediksi yang akurat.(Mohri,2018) Definisi pengalaman disini ialah informasisebelumnya yang telah tersedia dan bisa dijadikan data pembelajar. Dalam pembelajaran machine learning, terdapat skenario-skenario seperti :

1. *Supervised Learning*

   Penggunaan skenario supervised learning, pembelajaran menggunakan masukan data pembelajaran yang telah diberi label. Setelah itu membuat prediksi dari data yang telah diberi label.

2. *Unsupervised Learning*

   Penggunaan skenario unsupervised learning, pembelajaran menggunakan masukan data pembelajaran yang tidak diberi label. Setelah itu mencoba untuk mengelompokan data berdasarkan karakteristik-karakteristik yang ditemui.

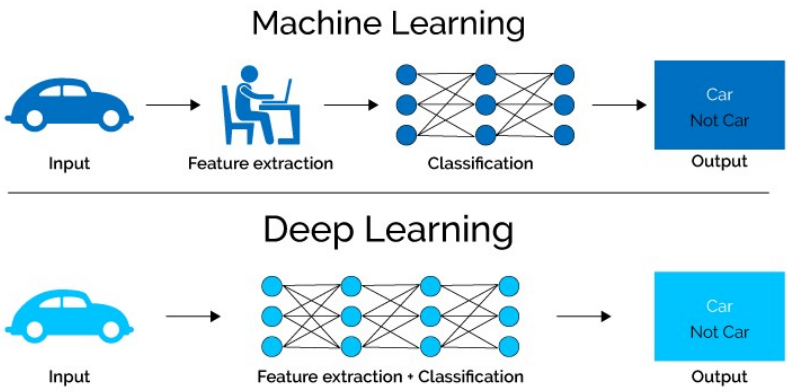
3. *Reinforcement learning*

   Pada skenario reinforcement learning fase pembelajaran dan tes saling dicampur. Untuk mengumpulkan informasi pembelajar secara aktif

dengan berinteraksi ke lingkungan sehingga untuk mendapatkan balasan untuk setiap aksi dari pembelajar.

2.5 Deep Learning

Pada dasarnya *deep learning* merupakan bagian dari *machine learning*. *Deep learning* adalah struktur algoritma yang terinspirasi dari struktur saraf otak yang disebut neuron, maka dari itu *deep learning* dapat disebut sebagai *artificial neural network.* Deep neural network atau disebut deep learning adalah pengembangan dari model machine learning neural networks, yaitu mengintegrasikan proses ekstraksi fitur dengan menambahkan *beberapa hidden layer*.(Zhang, et al, 2010) Teknik deep learning memberikan arsitektur yang sangat kuat untuk *supervised learning*. Dengan menambahkan lebih banyak lapisan maka model pembelajaran tersebut bisa mewakili data citra berlabel dengan lebih baik.



Gambar 2.5 *Konsep deep learning dan Machine learning*

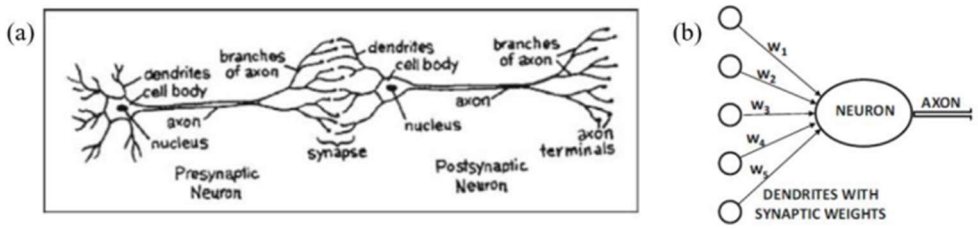
Penggunaan *deep learning* masa kini banyak aplikasinya, mulai dari untuk memproses gambar, video, suara, bahkan teks. Pada tahun 2010 penggunaan komputer untuk mengalahkan pemain Go profesional masih tidak mungkin. Lalu pada tahun 2020 seiring berkembangnya komputer dan *deep learning*, pemain Go profesional dikalahkan oleh sistem AlphaGo yang dibuat oleh DeepMind. Berkembangnya *deep learning* diakibatkan karena kemajuan teknologi dalam menyimpan data yang besar serta kapasita komputasi yang besar untuk melakukan pengolahan terhadap data yang banyak juga.

2.6 Neural Network

Neural Network adalah salah satu teknik *machine learning* yang populer dengan mensimulasikan mekanisme pembelajaran yang terinspirasi dari bagaimana cara sistem saraf makhluk biologis. Sistem saraf terdiri dari sel yang disebut dengan neuron yang saling terhubung

menggunakan *axon* dan *dendrites* dengan penghubung antara keduanya yang disebut sinapsis (Aggarwal, 2018).

Neuron terdiri dari tiga komponen utama, yaitu dendrit, badan sel (*cell body*), dan *axon*. Informasi yang diterima dari luar otak akan memiliki bentuk berupa sinyal listrik pada saraf. Dendrit menerima sinyal listrik dari neuron untuk kemudian dikirim ke 8 badan sel, yang berlanjut ke *axon* dan akhirnya dikirimkan ke neuron terdekat melalui sinapsis atau celah mikroskopik (Basheer & Hajmeer, 2000).



Gambar 2.6 *(a) Sistem saraf pada makhluk hidup, (b) arsitektur neural network*

Gambar 2.6 menunjukan ilustrasi perbandingan jaringan syaraf secara biologis dan jaringan *neural network*. Seperti pada sistem saraf, *neural network* juga terdiri dari unit-unit komputasi yang disebut sebagai *neuron. Neuron* tersebut dihubungkan oleh bobot, yang memiliki peran seperti sinapsis pada sistem saraf. Pada *neural network*, *input* dipetakan ke *output* secara langsung dengan fungsi linier. Model *neural network* yang paling sederhana disebut *single layer neural network,* dan pada umumnya *neural network* mempunyai komponen seperti *weight,* dan *activation function*.

2.6.1 Weight

Weight merupakan bagian dari *Artificial Neural Network* yang merubah nilai input data didalam *hidden layers* pada *neural network.* Nilai input dari layer sebelumnya akan dikali dengan *weight* sebelum akan diteruskan ke neuron selanjutnya pada *artificial neural network. Weight* akan diupdate pada saat proses *training. Weight* akan membantu *neural network* dalam menentukan kepentingan suatu fitur untuk menemukan solusi pada *neural network* dalam menghubungkan input dan output pada *neural network.*

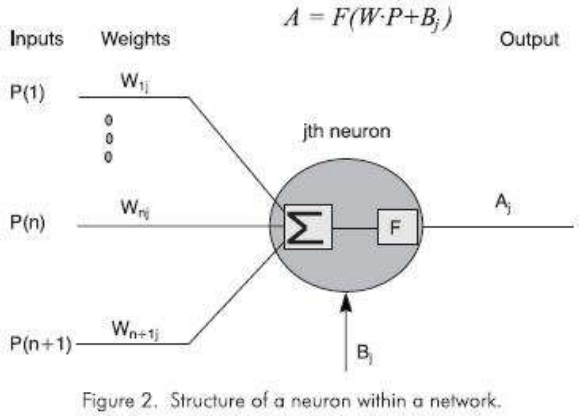


Figure 2. Structure of a neuron within a network.

Gambar 2. 7 *Ilustrasi nilai input dan weight pada neural network*

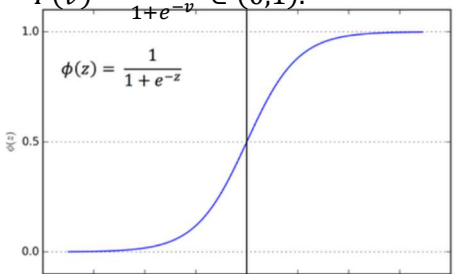
Pada gambar 2.7 nilai input x akan dikalikan *weight* dan ditambahkan dengan bias sehingga menghasilkan nilai y yang akan digunakan untuk neuron selanjutnya ataupun sebagai nilai output.

2.6.2 Activation Function

Berikut adalah beberapa fungsi aktivasi yang sering digunakan yaitu *sigmoid, hyperbolic tangent, rectified linear unit,* dan *softmax*.

1. Fungsi Sigmoid
   Fungsi sigmoid memiliki *range* nilai (0,1) sehingga dapat digunakan jika menginginkan *output* berupa probabilitas. Fungsi ini juga dapat diturunkan sehingga bisa digunakan untuk melatih model.

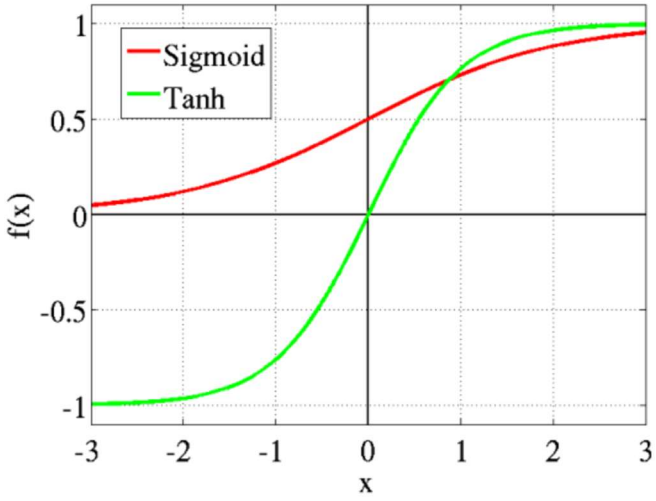
$$\Phi(v) = \frac{1}{1+e^{-v}} \in (0,1).$$



Gambar 2.8 *Grafik fungsi aktivasi sigmoid*

2. Fungsi *Hyperbolic Tangent*
   Fungsi *hyperbolic tangent* dapat didefinisikan sebagai rasio antara fungsi $sinh$ dan $cosh$.

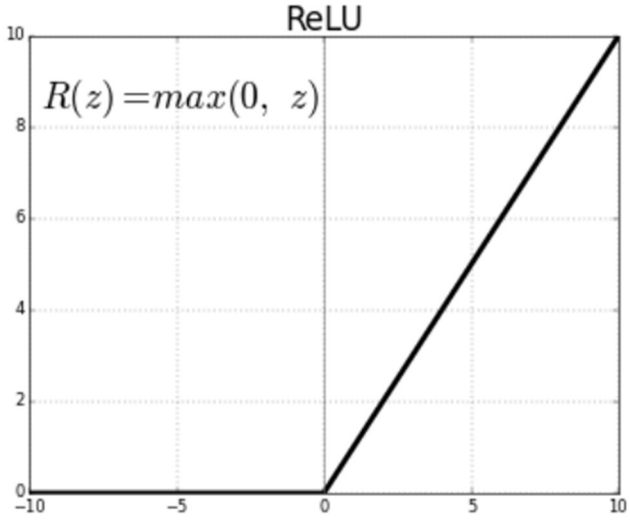
$$\Phi(v) = \tanh(v) = \frac{\sinh(v)}{\cosh(v)} = \frac{e^{2v}-1}{e^{2v}+1} \in (-1,1).$$



Gambar 2.9 *Perbandingan fungsi sigmoid dan tanh.*

3. Fungsi *Rectified Linear Unit* (ReLU)
   Fungsi ReLU tidak perlu menghitung fungsi eksponensial menyebabkan fungsi aktivasi ReLU lebih menguntungkan secara komputasional dari kedua fungsi aktivasi sebelumnya.

$$\Phi(v) = \max(0, x) \in [0, \infty). \quad (2.6)$$



Gambar 2.10 *Fungsi Basic Rectified Linear Unit*

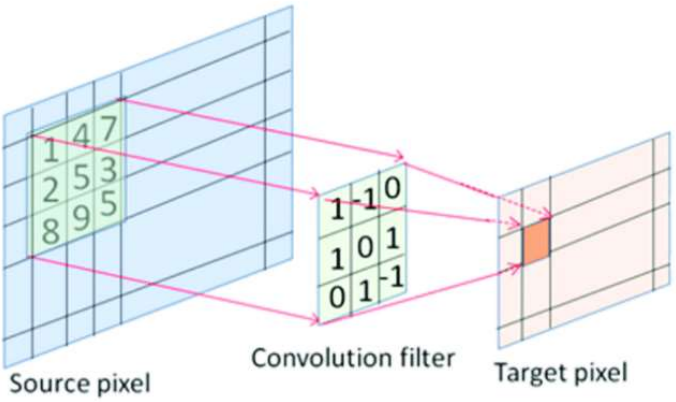
4. Fungsi *Softmax*

   Fungsi aktivasi *softmax* menerima *input* nilai *pre-activation* berupa vektor berdimensi-c $v = [v_1, v_2, \ldots, v_c]$ dengan tiap-tiap elemen v_i bernilai real.

$$\Phi(v) = \frac{\exp(v_i)}{\sum_{j=1}^{c} \exp(v_j)}, \forall i \in \{1, \ldots, c\}$$

2.7 Convolutional Neural Network

Convolutional Neural Network (CNN) adalah jenis neural network yang sering digunakan untuk aplikasi *deep learning* pada input gambar. Convolutional layer juga terdiri dari neuron yang tersusun yang berfungsi sebagai filter dengan panjang dan tinggi. CNN menggerakan sebuah konvolusi filter berukuran tertentu untuk mendapatkan fitur representatif baru dari sebuah gambar.

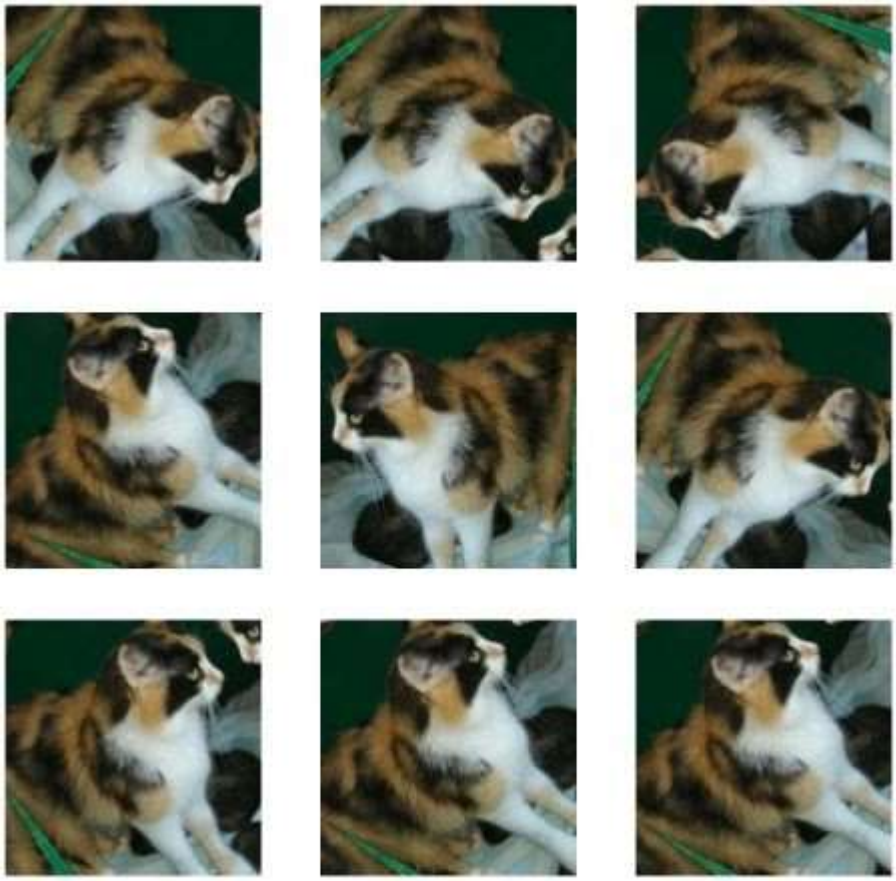


Gambar 2.11 Convolutional layer

Nilai dari tiap target pixel dapat disederhanakan lagi menjadi matriks yang lebih kecil biasanya didalam *pooling layer*. *Pooling layer* digunakan agar suatu matriks input dapat lebih sederhana serta mengekstraksi fitur yang penting pada suatu input contohnya seperti gambar. Pada aplikasi klasifikasi biasanya di layer akhir digunakan *neural network* untuk memprediksi apakah gambar tersebuh masuk ke kelas yang sudah dibuat.

2.8 Augmentasi Data

Augmentasi data merupakan sebuah metode untuk memanipulasi data agar menambah variasi pada dataset yang sudah ada. Jumlah data yang dinilai kurang jumlahnya dapat membuat model *machine learning* kurang baik dalam mempelajari sebuah pola dan dapat membuat performa model terbatas dalam melakukan klasifikasi. Maka dari itu khususnya pada *computer vision*, augmentasi data digunakan untuk menambah variasi dari sebuah gambar. Metode yang dilakukan ada beberapa untuk augmentasi data gambar. Metodenya seperti membalikkan gambar secara vertikal maupun horizontal, merubah warna dengan parameter *hue, saturation, contrast,* atau *brightness.* Gambar juga dapat diputar dengan sudut yang bervariasi.



Gambar 2. 12 Contoh Augmentasi data gambar

2.8 *Image Classification Algorithm*

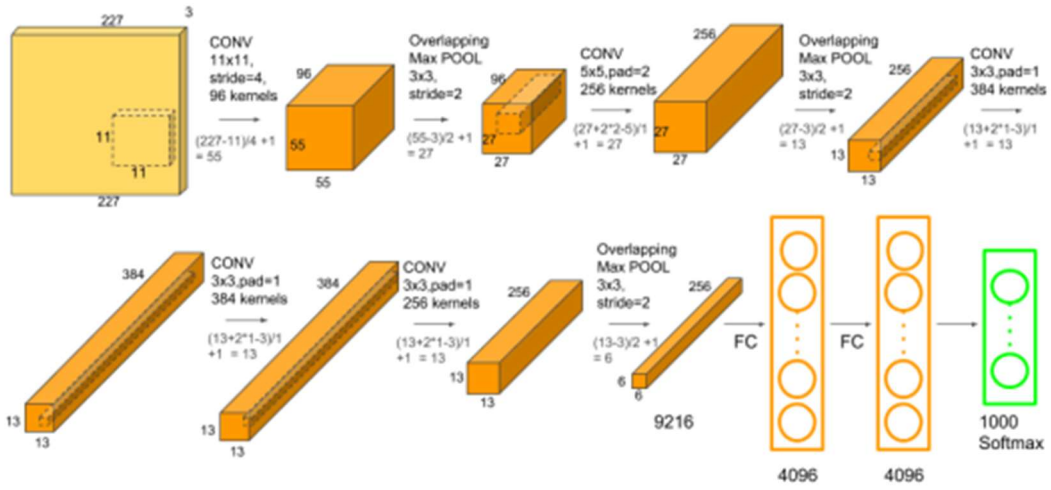
Penerapan *deep learning* untuk kasus-kasus yang menggunakan gambar sebagai input, atau pada penerapan *computer vision* banyak aplikasinya seperti untuk klasifikasi dan pendeteksian objek. Klasifikasi gambar digunakan untuk membuat komputer dapat mengetahui klasifikasi yang dilihat untuk digunakan sebagai bahan interaksi, khususnya seperti robot. Salah satu penerapan klasifikasi gambar pada robot yaitu digunakan sebagai data tambahan untuk lokalisasi suatu robot ketika melihat lingkungan sekitar.(Jin Taeseok, 2004).

Pengembangan algoritma *image classification* sudah dilakukan dari tahun 1989 dengan membuat jaringan perseptron yang menghubungkan tiap *pixel* pada gambar.(Yann LeCun et al., 1989). Namun penggunaan algoritma dengan menghubungkan tiap *pixel* gambar dianggap kurang efektif. *Convolutional neural network* merupakan jawaban agar pengolahan nilai-nilai yang dapat digunakan sebagai fitur dari gambar membantu performa untuk *image classification*. Algoritma-algoritma *image classification* yang sudah banyak digunakan seperti Alex Net, ResNet, MobileNet, dan lain-lain.

2.8.1 AlexNet

Arsitektur ini dikenalkan oleh Alex Krizhevsky pada tahun 2012 dalam "The 2012 ImageNet LSVRC-2012 Competition". Arsitektur ini kemudian banyak dikenal sebagai AlexNet sesuai dengan nama penciptanya. AlexNet berfokus pada pengembangan metode machine learning yang mana dikatakan bahwa hal ini dapat tercapai dengan mengumpulkan banyak data, melatih model yang lebih kuat, dan menggunakan teknik yang lebih baik untuk mencegah overfitting (model mempelajari seluruh detail, dan noise pada data sehingga menimbulkan pengaruh negatif). AlexNet dirancang untuk mampu mengolah dataset berukuran besar.

Seperti yang terlihat pada gambar 2.10, jaringan ini terdiri dari delapan layer dengan beberapa pembobotan; lima layer pertama adalah convolutional, dan tiga sisanya adalah fully-connected (FC) layer. Keluaran dari layer FC terakhir terhubung pada softmax layer yang menghasilkan sebuah distribusi lebih dari 1000 kelas label. Adapula layer tambahan untuk melakukan normalisasi pada respon yang dihasilkan pada convolutional layer pertama dan kedua. Terdapat juga max-pooling layer yangdigunakan untuk mereduksi ukuran dan parameter terhubung pada response-normalization layer begitu juga pada convolutional layer kelima. Pada setiap convolutional layer dan fully-connected layer diaplikasikan fungsi aktivasi berupa ReLU non linear.

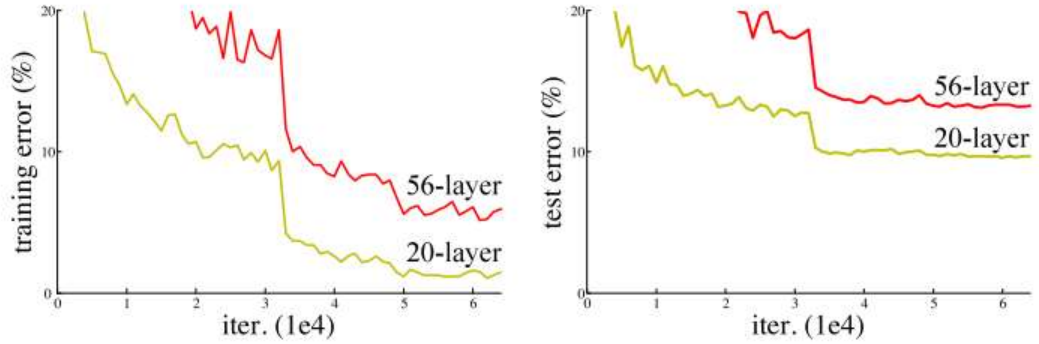


Gambar 2.12 Arsitektur AlexNet

2.8.1 ResNet

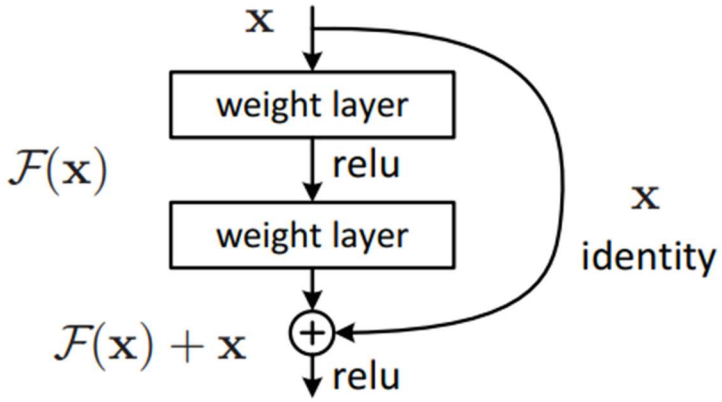
Kegunaan *Artificial Neural Network* dalam mendeteksi objek sudah dilakukan menggunakan arsitektur model *deep learning* yang umumnya berbasis *Convolutional Neural Network*. Sejak penemuan AlexNet kemajuan dalam *deep learning* untuk mendeteksi objek pada gambar semakin banyak dimulai dari VGG *network*, GoogleNet yang mempunyai lapisan jaringan semakin dalam atau *hidden layer* yang semakin banyak. Namun penambahan lapisan jaringan atau

menggunakan *hidden layer* yang banyak tidak membuat sebuah model mempunyai performa yang lebih baik.



Gambar 2.13 Performa model dengan 20 layer dan 56 layer (Kaiming He , et al)

Pada gambar 2.13 terdapat grafik *training error* dan *test error* terhadap iterasi untuk dua model yang mempunyai *hidden layer* 20 dan 56. Pada grafik tersebut dapat dilihat bahwa penambahan jumlah *hidden layer* tidak membuat performa model lebih baik. Hidden layer berjumlah 20 dapat mempunyai error lebih kecil dibanding dengan model berjumlah hidden layer berjumlah 56 dikarenakan parameter yang semakin banyak dengan bertambahnya hidden layer membuat sebuah model lebih susah dalam meng-*generalisasi* ResNet diciptakan sebagai solusi dalam mengembangkan performa model *machine learning*. ResNet menggunakan teknik *skip* yang dilakukan pada bagian *residual block*. Bagian yang terhubung oleh *residual block* akan melewatkan *layer* tertentu agar nilai pada neuron sebelumnya tidak di*train* pada *layer* yang dilewati.



Gambar 2.14 Residual Block untuk skip

Kelebihan menggunakan teknik *skip* berikut agar ketika ada *layer* yang membuat performa memburuk maka akan dilewati dengan *regularisasi*. Teknik yang mirip dengan cara kerja ResNet adalah "highway networks" yang menggunakan teknik *skip* namun tidak mempunyai performa yang lebih baik dari ResNet.

2.9 Evaluation Metrics

Evaluation metrics atau metrik evaluasi merupakan cara yang digunakan kualitas dari suatu *machine learning model*. Ada banyak metrik evaluasi yang digunakan untuk mengukur fungsi suatu *machine learning model.* Pada tiap aplikasi *machine learning* maupun *deep learning* digunakan metrik evaluasi yang berbeda-beda. Pada kasus mengklasifikasikan objek dapat digunakan metrik akurasi, *precision, recall,* nilai F1, dan masih banyak lagi. Pada kasus regresi dapat digunakan *mean absolute error, mean squared error, R-squared*. Pada aplikasi *image classification* juga sering digunakan evaluasi metrik seperti *precision, recall,* dan akurasi.

2.9.1 Precision

Precision merupakan evaluasi metrik yang digunakan untuk rasio *true positive* dari semua hasil klasifikasi positif. *Precision* menjelaskan seberapa presisi model dari dalam mengklasifikasi yang sesuai dengan kelas aslinya. Persamaan nilai *Precision* dapat dilihat sebagai berikut.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$
$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

Pada persamaan *Recall* dapat dilihat ketika nilai *false positive* naik maka nilai *precision* akan menurun. Nilai *recall* dapat dijadikan acuan seberapa benar model dalam mengklasifikan kelas tertentu.

2.10 Penelitian terdahulu

2.10.1 Traffic light detection and recognition for autonomous vehicles (Guo Mu et al. 2015)

Guo Mu dan kawan-kawan melakukan penelitian untuk mendeteksi lampu lalu lintas menggunakan algoritma berbasis kamera yaitu dengan merubah RGB color space menjadi *hue-saturation-value* (HSV) lalu difilter dan untuk mencari bounding boxes menggunakan algoritma *histogram of oriented gradients*(HOG) sebagai algoritma pendeteksi dan diklasifikasikan menggunakan *Support Vector Machine*(SVM). Penelitian dilakukan menggunakan mobil Hyundai Tucson yang sudah dimodifikasi menggunakan sensor lidar, dengan kamera yang terpasang pada depan mobil serta di kiri dan kanan bagian mobil juga.

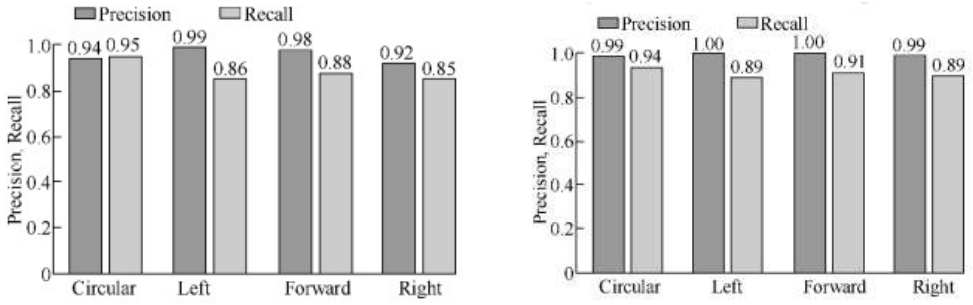
Gambar 2.15 Mobil Hyundai Tucson yang sudah dimodifikasi

Penelitian dilakukan menggunakan 13 kelas untuk identifikasi yaitu lampu lingkaran, lampu panah kanan, lampu panah kiri, lampu panah lurus, lampu panah kanan dengan warna hijau, merah, dan kuning. Eksperimen dilakukan menggunakan komputer Intel i7 quad-core dan dilakukan beberapa waktu yang berbeda. Hasil dari penelitian berikut.

Type	Correct	Missing	False alarm
Circular red	684	35	43
Circular green	921	64	14
Left arrow red	102	17	1
Left arrow green	184	22	0
Forward arrow red	84	12	2
Forward arrow green	204	19	1
Right arrow red	34	6	3
Right arrow green	264	31	2

Tabel 2.1 *Tabel hasil penelitian Guo Mu et al.*

Pada penelitian ini didapatkan nilai *precision* dan recall pada bentuk serta warna tertentu. Dapat dilihat sebagai berikut.

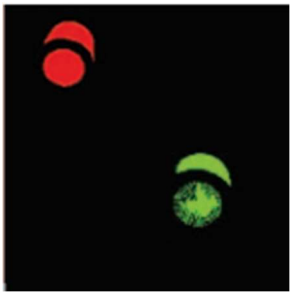


(a) (b)

Gambar 2. 16 Grafik precision dan recall saat (a) lampu berwarna merah dan (b)saat lampu berwarna hijau

2.10.2 Real-Time Detection and Classification of Traffic Light Signals (Said A. F., Et al. 2016)

Pada penelitian ini penulis membuat suatu algoritma untuk mengklasifikasi lampu lalu lintas dan mendeteksinya. Salah satu metode yang digunakan adalah dengan memanfaatkan *color space* sebagai fitur. Lalu dilakukan cara Lookup Table untuk beradaptasi dengan keadaan pencahayaan pada lingkungan. Dengan didapatkannya bagian lampu lalu lintas setelah dilakukan *color segmentation* dapat dilakukan klasifikasi menggunakan filter agar dapat mendeteksi warna pada lampu lalu lintas. Pada penelitian ini didapatkan hasil nilai presisi dan *recall* sebesar 95% dan 94,7%.



Gambar 2.17 Hasil Color Segmentation

2.10.3 Deep Convolutional Traffic Light Recognition for Automated Driving (Bach M, et al. 2018)

Pada penelitian ini Martin Bach dan kawan-kawan membuat algoritma yang memodifikasi arsitektur Faster R-CNN untuk mendeteksi serta mengklasifikasi status lampu lalu lintas pada jalanan. Dataset yang digunakan pada penelitian ini adalah DriveU Traffic Light Dataset. Peneliti memodifikasi arsitektur Faster R-CNN pada bagian *feature extractor* menjadi menggunakan 50 layer ResNet, pada Faster R-CNN awalnya menggunakan VGG net. Serta pada blok *conv4* pada layer ResNet dijadikan sebagai *shared feature layer* untuk *region proposal*. Serta pengurangan jumlah layer dari 16 step menjadi 8 step pada bagian residual. Pada bagian *box classifier* menggunakan *convolutional layer, average pooling layer*, dan *fully connected layer.* Kelas label yang digunakan untuk mendeteksi lampu lalu lintas dibagi sebagai beberap kelas yaitu hijau, orange, merah, merah-orange, dan mati, dengan total 5 kelas. *Dataset* yang digunakan mempunyai gambar sebanyak 25.000 dengan resolusi tinggi. Penelitian dilakukan menggunakan *framework* TensorFlow Object Detection API. Hasil penelitian ditunjukan dengan nilai *average precision*(AP), dan AP normalisasi di tiap kelas serta pada besaran lampu lalu lintas pada gambar.

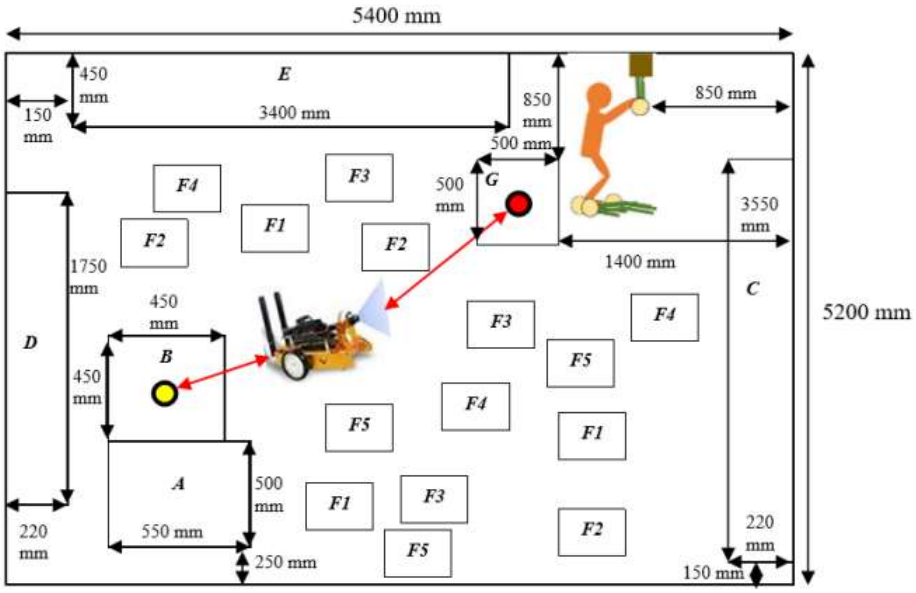
		Det	States					Types			
			red	green	off	amber	red-amber	circle	straight	left	right
all	AP	**0.83**	0.78	0.84	0.01	0.72	0.71	0.76	0.50	0.62	0.32
	AP_N	0.69	0.73	0.77	0.09	0.74	0.76	0.52	0.52	0.62	0.40
$0-5$ px	AP	0.00	0.00	0.00	-	-	-	0.00	0.00	-	-
	AP_N	0.00	0.00	0.02	-	-	-	0.00	0.00	-	-
>5 px	AP	0.84	0.80	0.85	0.02	0.73	0.72	0.77	0.51	0.63	0.33
	AP_N	0.71	0.75	0.78	0.10	0.75	0.77	0.53	0.53	0.63	0.42
$6-8$ px	AP	0.23	0.14	0.27	-	0.14	0.08	0.17	0.01	0.05	0.00
	AP_N	0.19	0.20	0.29	-	0.35	0.33	0.16	0.05	0.13	0.05
>8 px	AP	**0.92**	**0.91**	**0.93**	**0.02**	**0.78**	**0.77**	**0.84**	**0.59**	**0.73**	**0.36**
	AP_N	0.80	0.87	0.87	0.10	0.81	0.83	0.61	0.62	0.74	0.45
$8-10$ px	AP	0.66	0.60	0.67	-	0.40	0.39	0.47	0.06	0.23	0.00
	AP_N	0.63	0.74	0.72	-	0.57	0.83	0.49	0.20	0.47	0.12
>10 px	AP	0.93	0.92	0.94	0.02	0.83	0.77	0.85	0.65	0.77	0.39
	AP_N	0.84	0.90	0.89	0.12	0.86	0.83	0.64	0.69	0.79	0.49
$10-12$ px	AP	0.77	0.70	0.79	-	0.59	0.56	0.53	0.17	0.39	0.01
	AP_N	0.77	0.85	0.84	-	0.81	0.83	0.59	0.43	0.68	0.10
>12 px	AP	0.93	0.92	0.94	0.02	0.83	0.76	0.84	0.68	0.77	0.44
	AP_N	0.85	0.91	0.90	0.13	0.87	0.83	0.65	0.73	0.81	0.55

Tabel 2.2 Tabel hasil percobaan Martin Bach dan kawan-kawan

Pada penelitian ini didapatkan bahwa semakin besar lampu lalu lintas pada gambar *input* tentu nilai AP nya semakin tinggi. Serta model juga kesulitan dalam mendeteksi lampu yang tidak menyala. Pada penelitian ini penulis menduga bahwa *false positive* kebanyakan akibat lampu merah bertanda pejalan kaki, maka dari itu penambahan kelas untuk lampu lalu lintas pejalan kaki akan menambah performa model lebih baik lagi.

2.10.4 Deep Learning-Based Self-Driving Car: JetBot with NVIDIA AI Board to Deliver Items at Agricultural Workplace with Object-Finding and Avoidance Functions(Kawakura S, et al. 2020)

Penelitian berikut dilakukan oleh Shinji Kawakura dan kawan-kawan untuk membuat robot yang dapat mencari serta menghindar dari halangan berupa benda-benda yang biasanya ada di lingkungan kerja agrikultur. Objek-objek yang akan dihindari adalah sarung tangan, sepatu boot, sabit, gunting, dan cangkul. Peneliti menggunakan alat NVIDIA JetBot yang merupakan robot kecil menggunakan komputer NVIDIA Jetson Nano. Penelitian ini dilakukan dalam beberapa tahap yaitu, pertama tahap mendesain dan memvalidasi sistem. Kedua, perakitan serta penyetelan hal-hal minor pada sistem. Ketiga, pengumpulan data rintangan. Keempat, melakukan eksekusi pada model *deep learning*. Kelima, melakukan percobaan pada gudang *indoor* sesuai dengan keadaan saat kerja sehari-hari. Keenam, memproses data hasil percobaan. Percobaan dilakukan menggunakan model YOLO, khususnya YOLOtiny versi 2, dengan menggunakan Non-Max Suppression. Total gambar yang dikumpulkan oleh peneliti sebanyak 100 gambar, 20 gambar setiap rintangan. Percobaan dilakukan dengan menjalankan JetBot pada titik B ke titik H, dan juga sebaliknya. Percobaan dilakukan sebanyak 5 kali tiap jalur.



Gambar 2.18 Jalur Percobaan JetBot

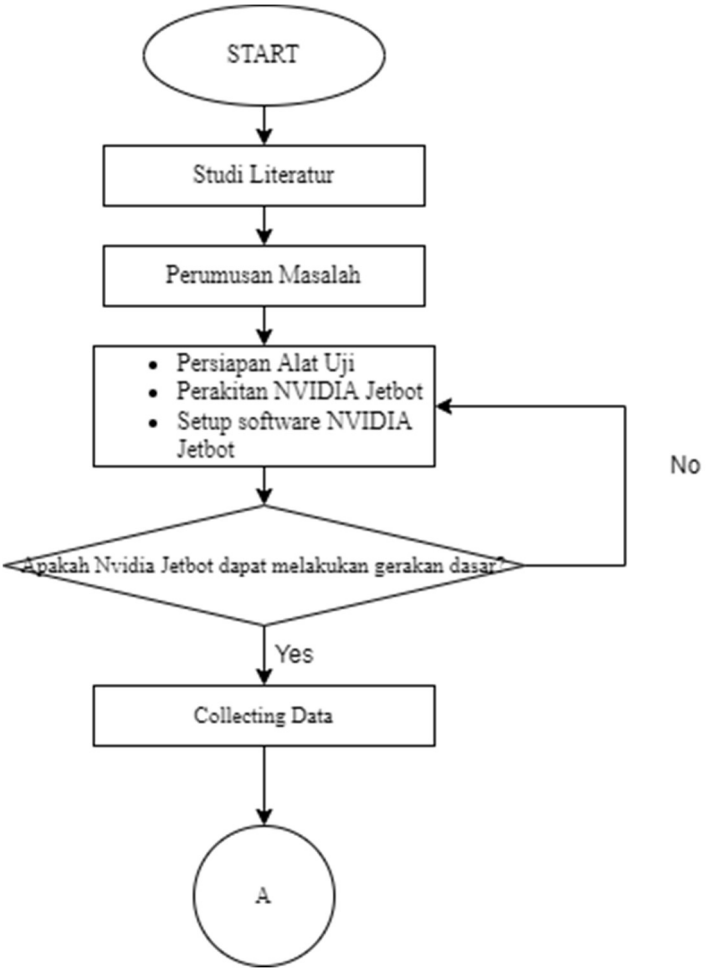
Dari penelitian yang dilakukan didapatkan tingkat keberhasilan 56% dan 60% dari dua lintasan yang divariasikan. Pada percobaan ini juga ditemukan beberapa fenomena yaitu JetBot tidak mendeteksi rintangan dengan cepat sehingga tidak dapat menghindar, JetBot terkadang juga mendeteksi objek terlalu cepat.

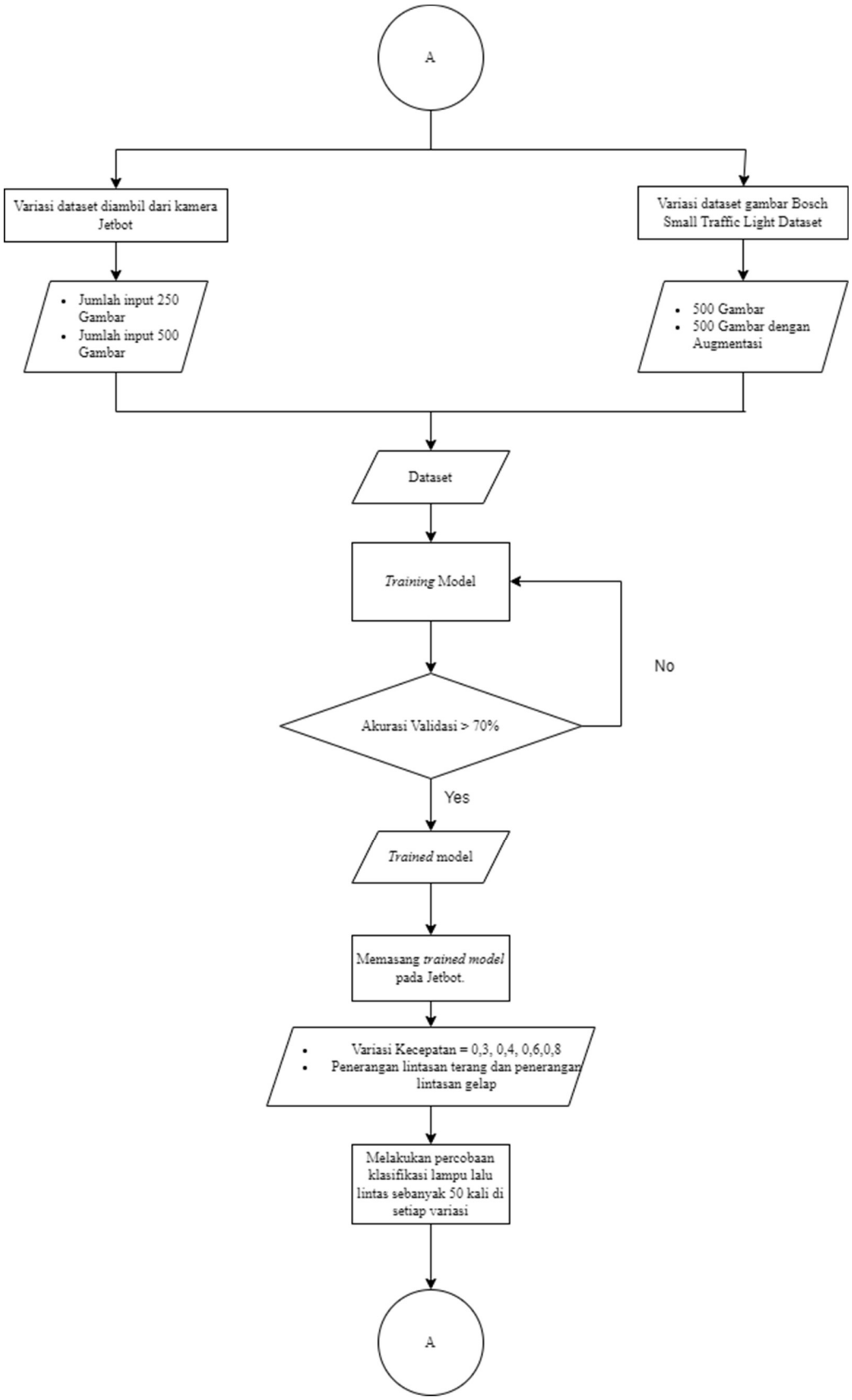
"Halaman ini sengaja dikosongkan"

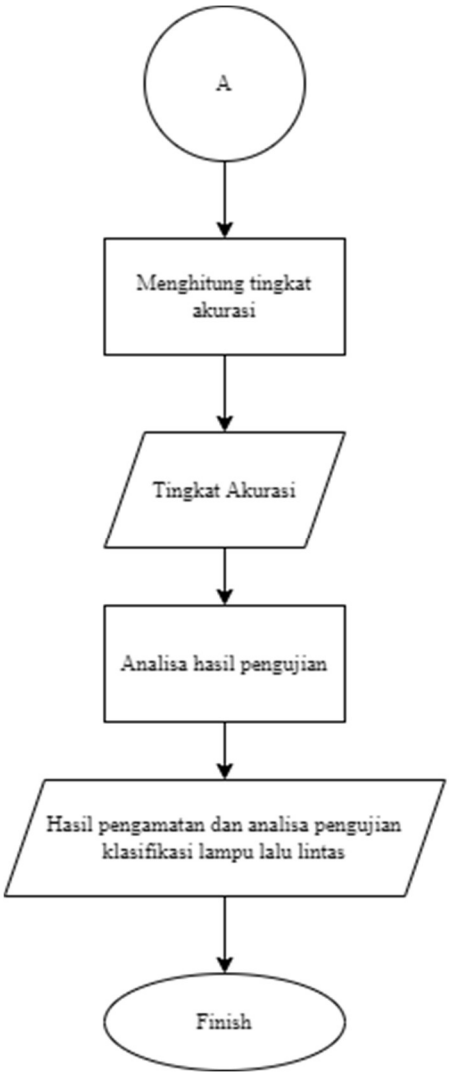
BAB III
METODOLOGI PENELITIAN

3.1 Metode Penelitian

Penelitian dilakukan dengan metode simulasi dengan validasi tingkat keberhasilan Jetbot dalam menjalankan fungsi pendeteksi lampu lalu lintas. Penelitian ini dilakukan dalam beberapa tahapan utama yaitu terdiri dari studi literatur, persiapan alat uji, *collecting data, training* model, evaluasi *trained* model, *deploying* model pada Jetbot , melakukan uji eksperimen dan pengambilan data dari uji Keberhasilan dan akurasi dan uji keberhasilan dengan variasi kecepatan, pengolahan data dan pembuatan grafik akurasi. Simulasi dilakukan dengan alat Jetbot yang sudah dirakit. Penelitian dijelaskan dalam beberapa langkah menggunakan diagram alir berikut.




A
Variasi dataset diambil dari kamera Jetbot
Variasi dataset gambar Bosch Small Traffic Light Dataset
Jumlah input 250 Gambar
Jumlah input 500 Gambar
500 Gambar
500 Gambar dengan Augmentasi
Dataset
Training Model
No
Akurasi Validasi > 70%
Yes
Trained model
Memasang trained model pada Jetbot.
Variasi Kecepatan = 0,3, 0,4, 0,6,0,8
Penerangan lintasan terang dan penerangan lintasan gelap
Melakukan percobaan klasifikasi lampu lalu lintas sebanyak 50 kali di setiap variasi
A



Gambar 3.1 Diagram Alir Penelitian

3.2 Studi Literatur

Pada tahap ini dilakukan pencarian referensi sebagai pendukung penelitian ini. Referensi tersebut dalam bentuk teori-teori dari penelitian yang berkaitan, pendapat para ahli serta data-data yang mendukung. Referensi akan digunakan sebagai dasar dan pendukung dalam melakukan eksperimen.

3.3 Persiapan Alat Uji

3.3.1 Alat

Alat yang digunakan pada simulasi ini adalah sebagai berikut:

1. Jetbot

Gambar 3.2 Jetbot Waveshare

- Merk : Waveshare
- Type : Waveshare JetBot AI Kit
- Controller : Jetson Nano
- Operating system : Ubuntu 18.04 LTS
- Program Language : Python
- Camera : 8MP 160° FOV wide angle camera
- Display : 0.91 inch OLED, 128×32 pixels
- Wireless NIC : 2.4GHz / 5GHz dual mode WIFI + Bluetooth 4.2
- Battery Solution : 12.6 V, 18650 batteries with protection
- Motor : TT motor; reduction rate 1:48; Idle speed 240RPM
- Chassis : Aluminium alloy chassis
- Teleoperation : gamepad, webpage
- Communication : WIFI
- Protection : over-current protection, voltage monitoring

2. Lampu Lalu Lintas miniatur dengan Arduino sebagai controller.

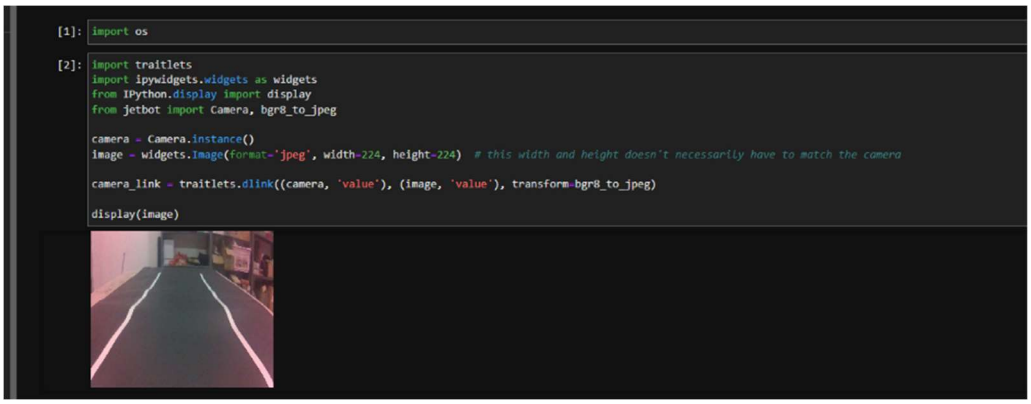
Gambar 2. 19 Miniatur lampu lalu lintas

3. Micro SD Card
4. Monitor eksternal
5. Kabel HDMI
6. Keyboard
7. Mouse
8. Lampu sorot
9. Jupyter Lab (Python 3)

3.3.2 *Collecting Data*

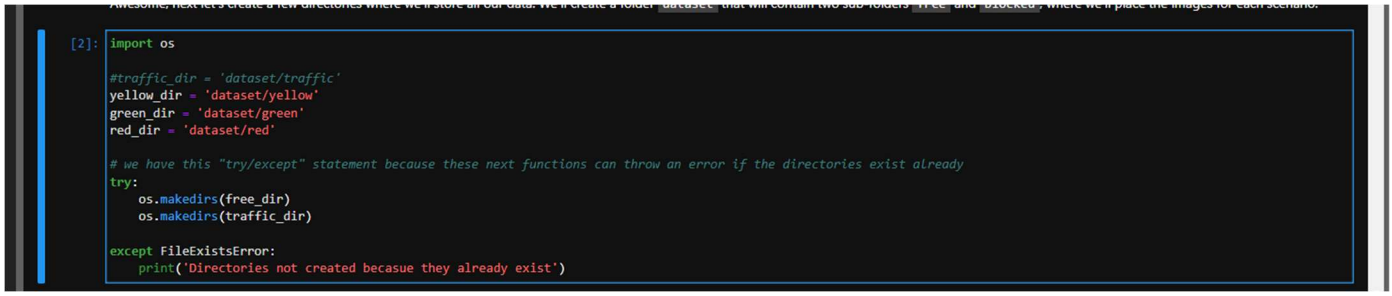
Pada penelitian ini pada tahap *Collecting data* adalah tahap pengumpulan data dari *dataset* yang sudah ada dari penelitian terdahulu dan *dataset* yang akan dibuat oleh penulis menggunakan NVIDIA Jetbot. *Dataset* yang digunakan penelitian ini adalah *Bosch Small Traffic Light Dataset.* (Behrendt dan kawan-kawan) Pada *dataset* ini digunakan 500 gambar lampu lalu lintas yang diambil oleh kamera pada dalam mobil dan diberi label warna lampu lalu lintas oleh peneliti publikasi. Pada *dataset* ini terdapat 3 kelas klasifikasi yaitu merah, kuning, dan hijau. *Dataset* yang dibuat oleh penulis dengan melakukan pengambilan data dengan kamera pada NVIDIA Jetbot, pengambilan data sebanyak 250 gambar dan 500 gambar. Berikut tahap-tahap melakukan *collecting data* untuk membuat data *training*.

1. Menghubungkan robot dengan computer yaitu dengan navigasi ke http://<jetbot_ip_address>:8888 dan melakukan *sign in* dengan Akun default password JetBot.
2. Klik folder dengan nama "*Notebooks*" dan setelah itu klik Kembali folder dengan nama "*Collision_Avoidance"*
3. Buka file dengan nama "data_collection.ipynb" dan mulai menjalankan eksekusi kode dengan modifikasi untuk menyesuaikan beberapa kelas.
4. Setelah menjalankan program kode, JetBot akan otomatis menyalakan kamera damenghubungkannya ke Komputer yang telah terhubung sehingga akan tertampil tangkapan kamera padaJupyterLab seperti gambar 3.3.



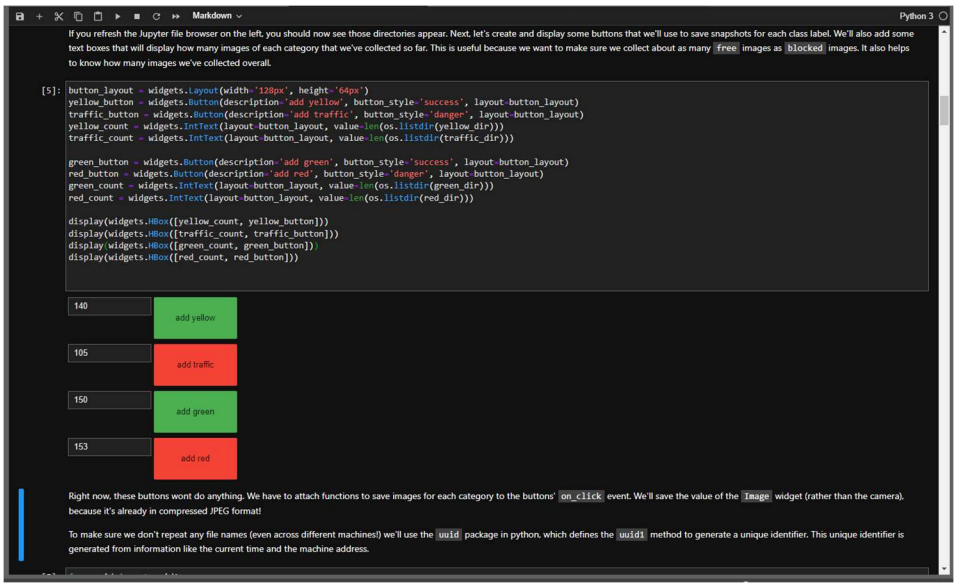
Gambar 3. 3 Aktivasi kamera untuk pengambilan data.

5. Membuat beberapa direktori tempat untuk menyimpan data yang telah diambil dengan membuat folder dari dataset yang berisi tiga sub-folder yaitu “green” “red”, dan “yellow” dimana tiap gambar akan disimpan dalam setiap scenario. Proses dapat dilihat pada Gambar 3.4.



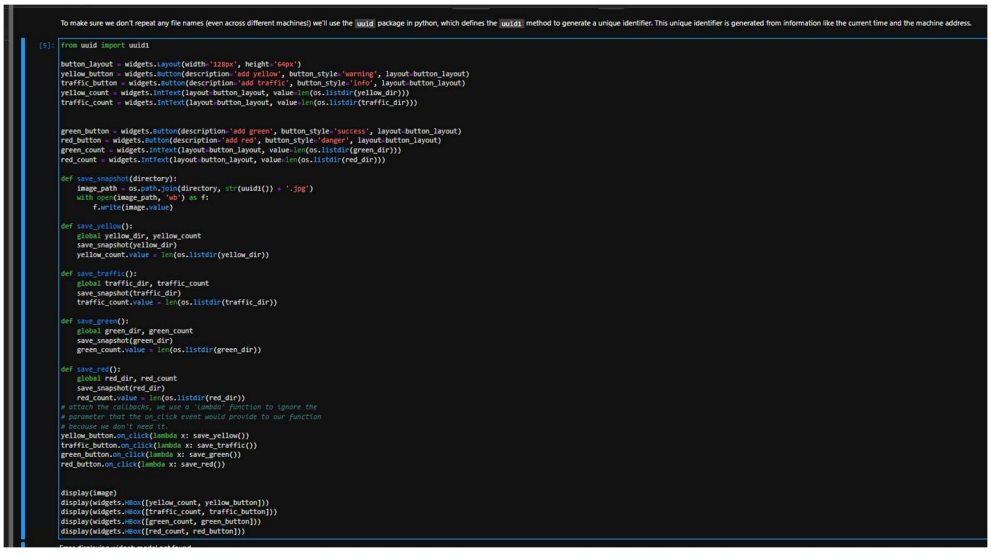
Gambar 3. 4 Pembuatan folder untuk data

6. Membuat dan menampilkan beberapa tombol yang akan digunakan untuk menyimpan snapshot untuk setiap label kelas. Ditambahkan beberapa kotak teks yang akan menampilkan berapa banyak gambar. dari setiap kategori yang telah dikumpulkan. Ini juga membantu mengetahui seberapa banyak gambar yang diambil secara keseluruhan.



Gambar 3. 5 Tombol untuk pengambilan data

7. Untuk membuat kedua tombol tersebut berfungsi, maka diperlukan melampirkan fungsi penyimpanan gambar untuk setiap kategori ke tombol "on_click" event. Nilai Widget gambar akan tersimpan dalamformat JPEG terkompresi. Untuk memastikan tidak ada nama file yang terulang, digunakan paket"uuid" di Python, yang dimana mendefinisikan metode "uuid1" untuk menghasilkan pengenal unik.Pengindentifikasi unik ini dihasilkan dari informasi seperti waktu saat ini dan alamat mesin. Dapatdilihat pada gambar 3.6.



Gambar 3. 6 Pembuatan fungsi untuk setiap tombol pengambilan data.

8. Mengambil data dengan gambar dimana kondisi warna lampu lalu lintas terdapat di depan JetBot. Pengambilan data dilakukan dengan merubah posisi

Jetbot agar memperbanyak data perspektif berbeda dalam melihat warna lampu lalu lintas.

9. Pengambilan gambar akan dilakukan dengan 250 dan 500 jumlah gambar untuk variasi dari kamera Jetbot.
10. Saat semua dataset sudah selesai diambil. Data tersebut dikompres menjadi satu file berbentuk zip. Dan setelah itu mendownload dan mengupload data tersebut ke GPU desktop untuk dilatih mengenai *traffic light classification*.

Gambar 3. 7 Sampel gambar pada dataset dari Jetbot

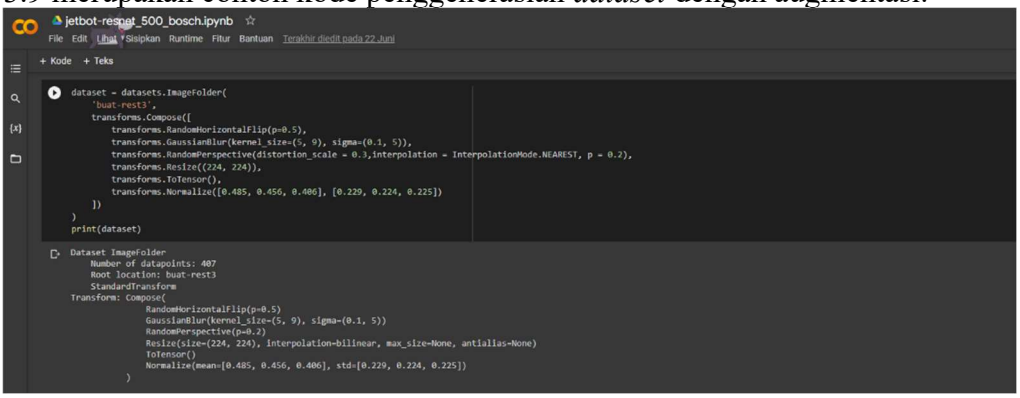
3.3.3 Training Model

Tahap *training model* merupakan tahap untuk melatih model agar dapat mendeteksi dan mengklasifikasikan warna lampu lalu lintas dengan benar. *Dataset* yang digunakan akan dibagi menjadi 3 bagian yaitu *train dataset, test dataset,* dan *validation dataset. Train dataset* merupakan bagian data yang akan digunakan untuk model *neural network* untuk mempelajari fitur-fitur yang terdapat pada gambar seperti bentuk lampu lalu lintas serta warna dari lampu lalu lintas tersebut dan membuat bobot yang sesuai pada model agar dapat mengklasifikasikan lampu lalu lintas dengan tepat. Model yang digunakan adalah model *deep learning* ResNet18 yang merupakan model dengan fitur skip network yang membuat akurasi model lebih baik dengan semakin banyaknya iterasi training model. Model yang didapat dari proses training ada 4 model terdiri dari model dengan *dataset* pengambilan penulis sebanyak 250 gambar, model dengan *dataset* pengambilan penulis sebanyak 500 gambar, model dengan *dataset* Bosch Small Traffic Light sebanyak 500 gambar, dan model dengan *dataset* Bosch Small Traffic Light sebanyak 500 gambar dengan augmentasi. Tahap-tahap pada *training* model adalah sebagai berikut:

1. Membuka jupyter notebook dan memanggil *library* serta fungsi-fungsi yang akan digunakan dengan syntax "import". Setelah melakukan pemanggilan *library* dan fungsi maka lakukan unzip untuk data yang akan digunakan sebagai data *training* model. Gambar 3.8 merupakan contoh kode untuk melakukan tahap ini.

Gambar 3. 8 Pemanggilan *library* dan fungsi serta *unzipping* dataset.

2. Tahap selanjutnya akan melakukan tahap penggenerasian *dataset* yang dapat digunakan pada fungsi *dataloader.* Penggenerasian *dataset* dari folder yang sudah dibagi dengan sub-folder setiap kelas klasifikasi dilakukan dengan fungsi datasets.ImageFolder. Pada tahap ini juga dapat dilakukan augmentasi data, untuk variasi data *training* 500 gambar Bosch Small Traffic Light Dataset dilakukan dengan memberi parameter *transform*. Parameter augmentasi yang digunakan pada tahap ini adalah RandomHorizontalFlip, GaussianBlur, RandomPerspective. Variasi tanpa augmentasi seperti variasi gambar dari Jetbot, dan Bosch Small Traffic Light Dataset tanpa augmentasi hanya diberikan parameter resize, toTensor, dan normalisasi. Gambar 3.9 merupakan contoh kode penggenerasian *dataset* dengan augmentasi.



Gambar 3. 9 Penggenerasian dataset dari folder gambar dengan augmentasi.

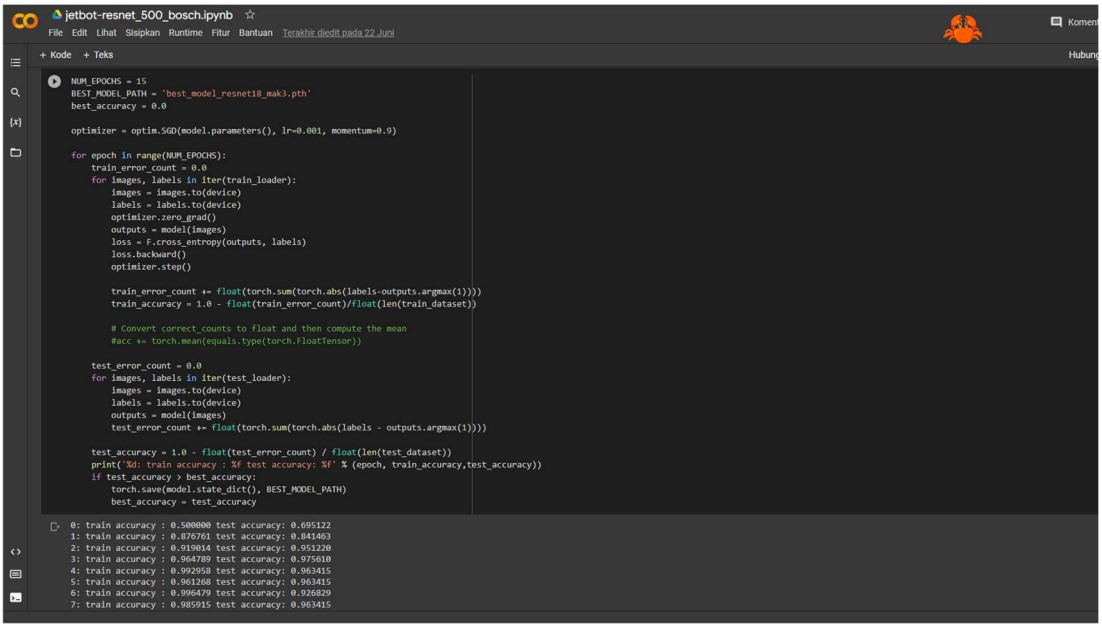
3. Tahap ini melakukan pembagian *dataset* menjadi 3 bagian yaitu, train, test, dan validasi. Presentase pembagian train, test, dan validasi secara berurutan adalah 70%, 20%, 10%. Hal ini dapat dilakukan dengan fungsi data.randomsplit. Contoh melakukan pembagian dataset dapat dilihat pada gambar 3.10

Gambar 3. 10 Pembagian dataset menjadi 3 bagian.

4. Selanjutnya akan dilakukan *loading* data dalam *batch* yang dapat digunakan untuk model *deep learning* belajar. *Loading* data dilakukan dengan fungsi DataLoader. Setelah itu akan dilakukan pembuatan variabel model serta membuat *layer* terakhir dari model disesuaikan dengan jumlah kelas yang akan diklasifikasi, pada penelitian ini adalah 3.

Gambar 3. 11 Proses *loading* data dan pemilihan model

5. Tahap ini akan dilakukan proses *training* model dengan parameter sebagai berikut, epochs berjumlah 15, dengan *optimizer* Stochastic Gradient Descent, dengan *learning rate* bernilai 0,001. Kode ini juga akan melakukan penyimpanan model. Penulisan kode dapat dilihat pada gambar 3.12.

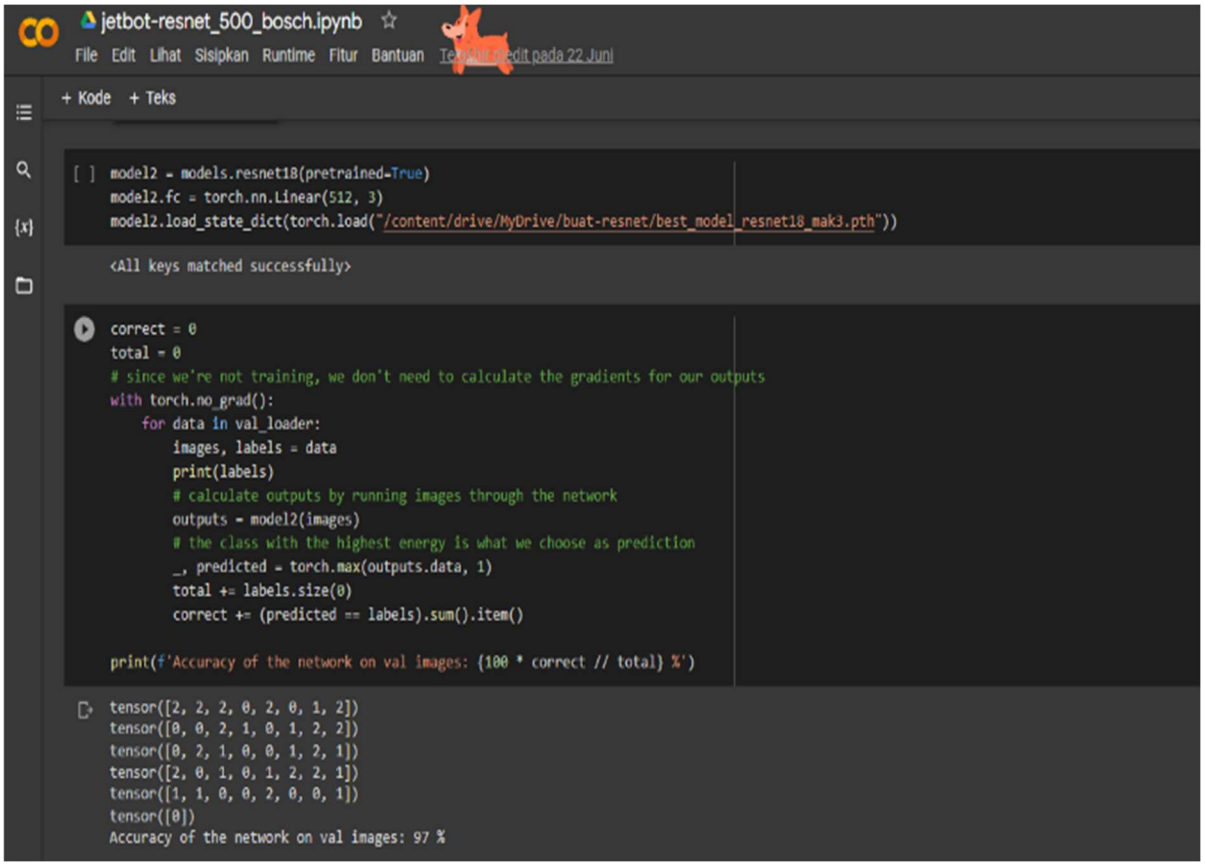


Gambar 3. 12 Kode untuk *training* model.

Pada proses ini juga didapatkan nilai akurasi pada test dataset sebagai acuan evaluasi model yang di-*train*.

3.3.4 Evaluasi *Trained* Model

Tahap evaluasi trained model adalah tahap saat melakukan evaluasi pada model yang sudah ditraining menggunakan metric yaitu akurasi. Tujuan mengevaluasi trained model ini untuk melihat performa model yang sudah dilatih dan diuji terhadap *test dataset*. Evaluasi dilakukan dengan menggunakan model yang sudah di-*training* dan melakukan klasifikasi terhadap gambar-gambar di *dataset* test dan *dataset* validasi yang sudah dilabeli sesuai kelasnya atau pada penelitian ini warnanya. Tahap evaluasi model terhadap *test* dataset yaitu terdapat pada proses ke-5 di sub-bab tahap *training* model. Evaluasi dengan *dataset* validasi juga dilakukan untuk melihat performa model yang sudah di-*train*. Perubahan parameter akan dilakukan apabila hasil evaluasi model yang sudah di-*train* terhadap *dataset* validasi tidak sampai nilai akurasi 70%.

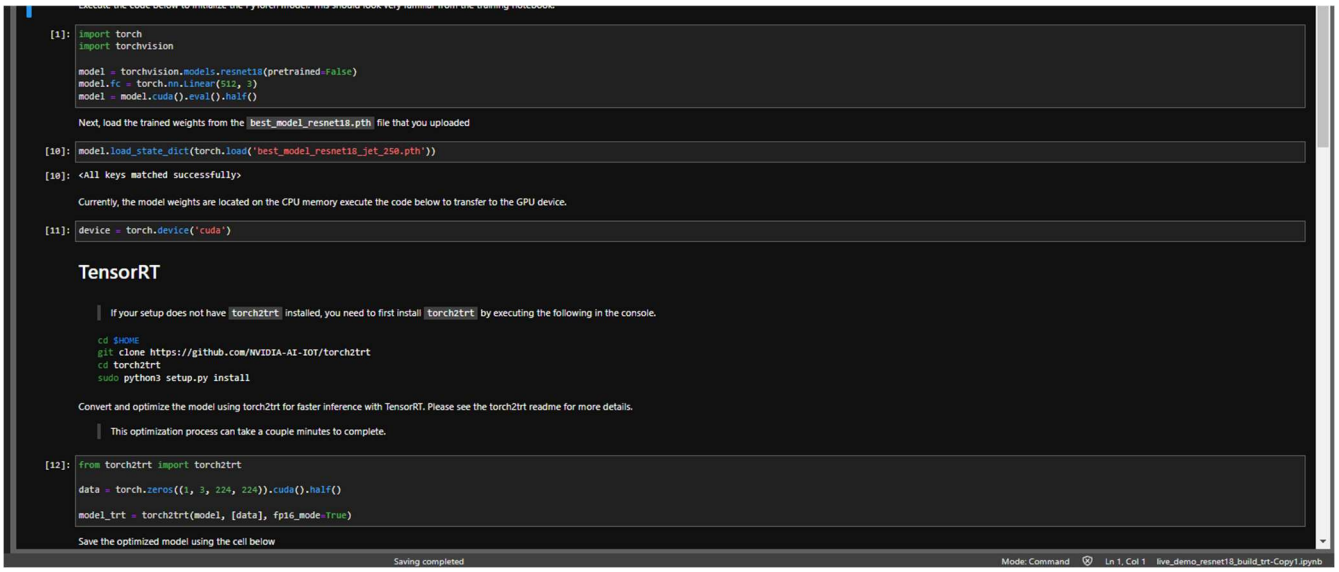


Gambar 3. 13 Kode untuk evaluasi model dengan *dataset* validasi

3.3.5 Pemasangan t*rained* model pada Jetbot

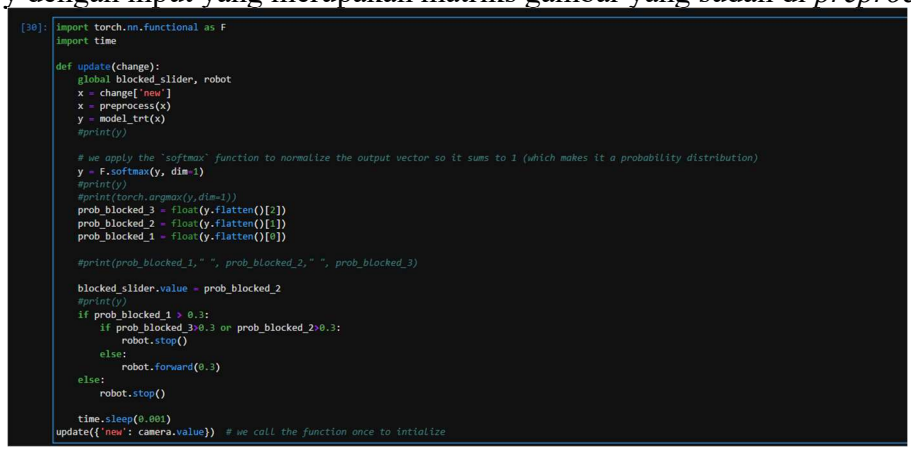
Pemasangan *trained model* pada jetbot atau model *deployment* saat model yang dievaluasi sudah mendapatkan performa yang baik yaitu bernilai akurasi diatas 70% dalam metrik akurasi maka model akan digunakan jetbot untuk mendeteksi lampu lalu lintas. Proses deployment adalah saat menggunakan model yang sudah dilatih agar dapat digunakan oleh Jetbot dengan menyesuaikan environment serta algoritma yang digunakan. Proses untuk melakukan *model deployment* adalah sebagai berikut:

1. Model akan dirubah ke bentuk TensorRT agar dapat memproses lebih ringan oleh GPU dari NVIDIA Jetbot.



Gambar 3. 14 Konversi model ke bentuk TensorRT

2. Lalu setelah model sudah menjadi bentuk TensorRT maka akan dimasukkan pada kode yang digunakan untuk memproses masukan gambar dari kamera dan dilakukan prediksi menggunakan model yang sudah dipasang. Pada kode dibawah ini dapat dilihat variabel y merupakan output dari model ResNet18 yang sudah dirubah pada bentuk TensorRT, variabel y dengan input yang merupakan matriks gambar yang sudah di *preprocess*.



Gambar 3. 15 Kode untuk melakukan klasifikasi serta pemberian respon gerakan dari Jetbot.

3.3.6 Uji Eksperimen dengan Variasi Kecepatan dan Variasi Pencahayaan.

Tahap uji eksperimen dilakukan dengan 2 pengujian yaitu uji eksperimen keberhasilan dan uji dengan variasi kecepatan. Uji eksperimen keberhasilan adalah pengujian Jetbot menggunakan model yang sudah di*train* dan di*deploy* ke Jetbot. Jetbot akan berjalan pada lintasan yang disiapkan dengan lampu lalu lintas miniatur dengan kecepatan tetap. Keberhasilan dilihat dari percobaan

sebanyak 50 kali setiap variasi uji dan Jetbot dapat berhenti saat lampu lalu lintas berubah dari warna hijau menjadi warna merah. Uji dengan variasi kecepatan adalah uji eksperimen Jetbot untuk mendeteksi lampu lalu lintas dan berhasil berhenti pada kecepatan yang berbeda. Uji variasi kecepatan dilakukan pada 4 kecepatan yang berbeda dan pengujian dilakukan 50 kali setiap variasi kecepatan. Variasi kecepatan yang akan dilakukan adalah 0.3, 0.4, 0.6, dan 0.8 pada *library* Jetbot dengan 2 variasi pencahayaan, yaitu saat terang dan gelap. Pencahayaan terang adalah dengan lampu ruangan dan kondisi pencahayaan gelap adalah dengan pencahayaan lampu senter seterang 50 lumens dari arah atas lintasan. Tingkat akurasi akan dihitung dari keberhasilan Jetbot merespon warna lampu lalu lintas dengan benar, dihitung dengan total Jetbot melakukan respon dengan benar terhadap warna lampu lalu lintas dibagi dengan total percobaan.

$$Akurasi = \frac{Total\ Percobaan\ Benar}{Total\ Percobaan}$$

Gambar 3. 16 Skema pengujian sederhana

“Halaman ini sengaja dikosongkan”

BAB IV
HASIL DAN PEMBAHASAN

4.1 Model CNN yang diperoleh dan Ujian Pendeteksian

Proses training model dilakukan dengan beberapa variasi input data seperti jumlah data, dan asal data. Model yang dihasilkan dapat dilihat hasilnya menggunakan besaran akurasi tes dan akurasi validasi. Adapun hasil akurasi tes dan validasi dari setiap variasi data input adalah sebagai berikut.

Tabel 4. 1 Hasil akurasi test dan validasi model yang sudah di-*training* untuk setiap dataset.

500 Gambar Bosch Small Traffic Light Dataset Augmented		
Epoch	Test	Validation
15	0.99	0.98

500 Data Bosch Small Traffic Light Dataset Non Augmented		
epoch	Test	Validation
15	1	0.96

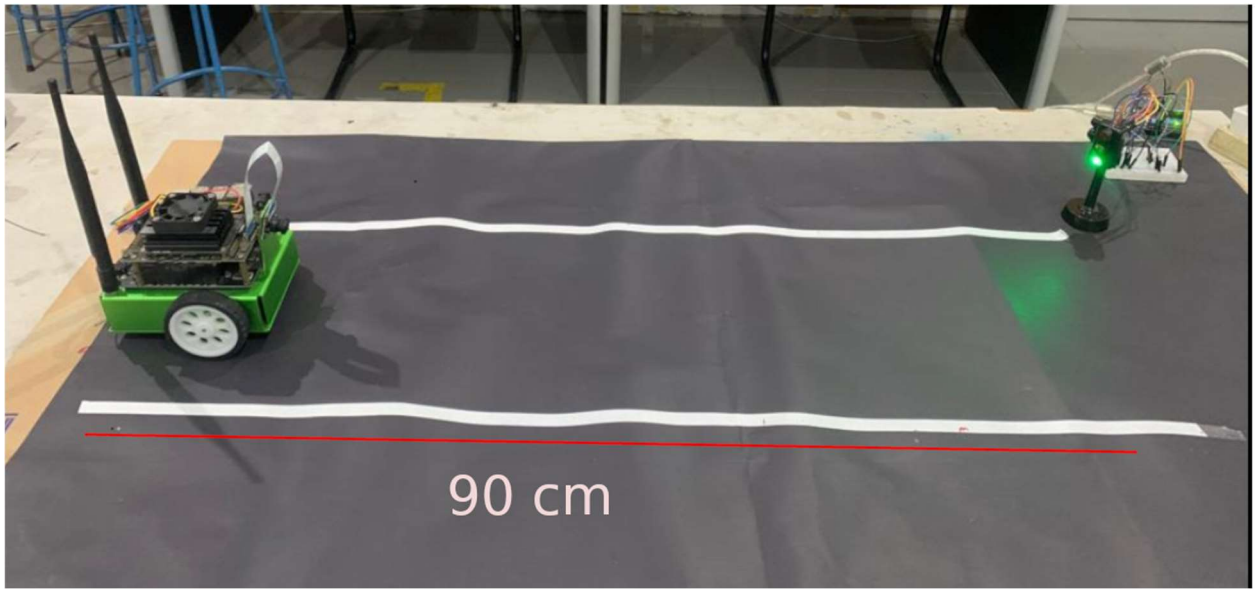
250 Gambar dari Kamera Jetbot		
Epoch	Test	Validation
15	1	0.96

500 Gambar dari Kamera Jetbot		
Epoch	Test	Validation
15	1	1

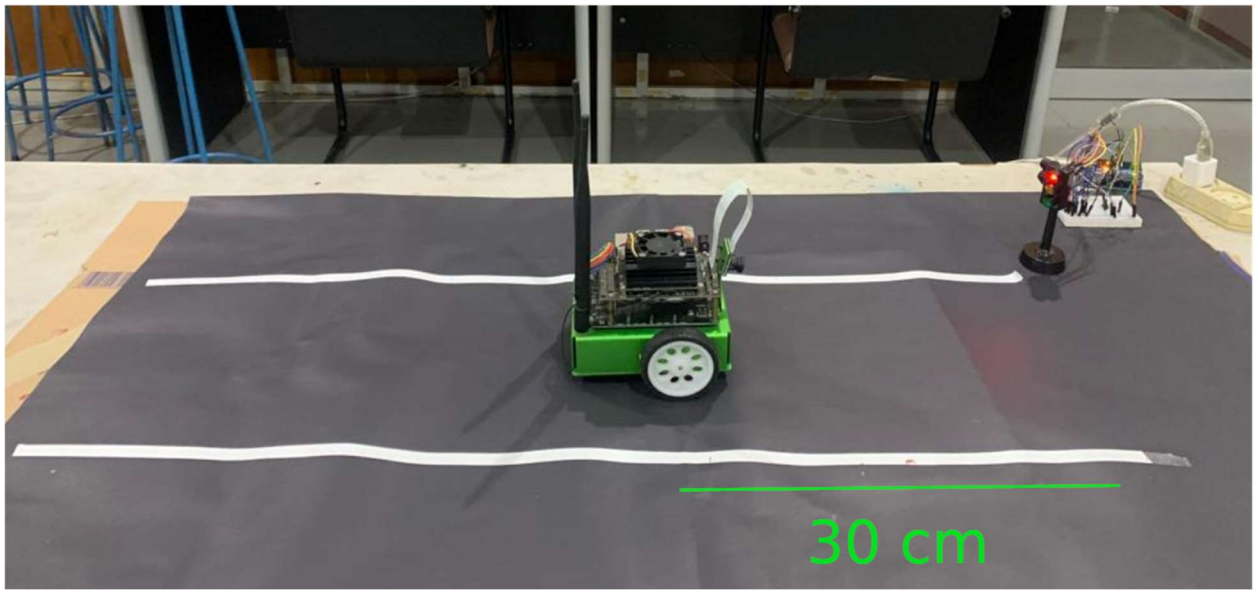
Pada hasil training model dapat dilihat dari tabel 4.1 dilihat nilai akurasi model sudah memenuhi syarat untuk penelitian ini yaitu minimal akurasi 0.7, baik pada proses validasi maupun tes. Hasil ini dapat dilihat model terjadi overfitting.

Ujian pendeteksian dilakukan dengan miniatur lampu lalu lintas dan lintasan sepanjang 90 cm, awal percobaan Jetbot akan diletakkan sejauh 90 cm dari lampu lalu lintas, sedangkan titik perubahan warna lampu dari warna hijau ke kuning berada pada jarak 30 cm dari lampu lalu lintas. Pengujian dianggap berhasil ketika Jetbot dapat mengklasifikasi warna lampu lalu lintas dengan tepat. Jetbot mengklasifikasi warna lampu lalu lintas dengan tepat saat lampu lal u lintas berwarna hijau maka Jetbot akan berjalan, saat lampu lalu lintas berwarna kuning Jetbot berhenti, dan saat

lampu lalu lintas berwarna merah Jetbot tetap berhenti di tempat. Setiap kegagalan yang dilakukan Jetbot dicatat. Jalur untuk pengujian Jetbot dapat dilihat pada gambar dibawah ini.



Gambar 4. 1 Jalur pengujian Jetbot dalam klasifikasi lampu lalu lintas.



Gambar 4. 2 Jalur pengujian Jetbot dalam klasifikasi lampu lalu lintas.

Pada pengujian pendeteksian lampu lalu lintas Jetbot akan menggunakan model machine learning yang di-train dari data training yang berbeda yaitu 500 data Bosch teraugmentasi, 500 data Bosch tanpa augmentasi, 250 data diambil dari Jetbot, dan 500 data diambil dari Jetbot. Lalu tiap pengujian model yang digunakan oleh Jetbot akan diuji pada kondisi penerangan yang berbeda dan

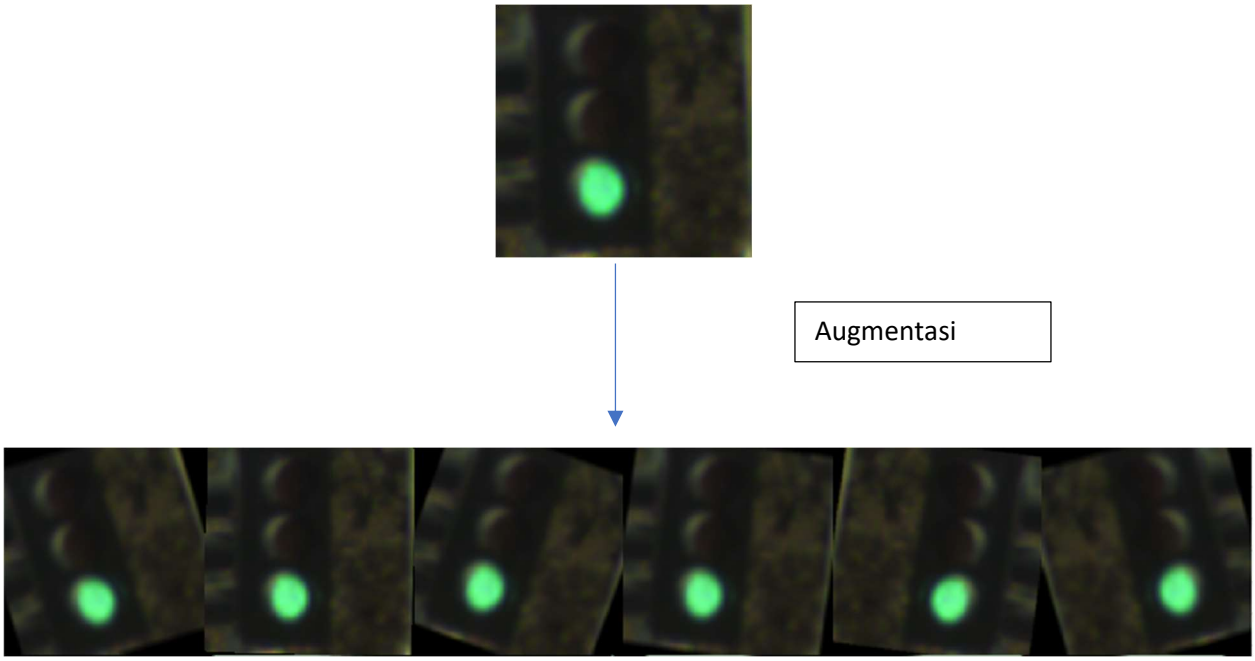
kecepatan yang berbeda. Untuk variasi kecepatan terdapat 4 variasi yaitu 0,3, 0,4, 0,6, dan 0,8 kecepatan pada modul Jetbot. Lalu pada variasi pencahayaan terdapat dua variasi pencahayaan yaitu variasi terang dengan pencahayaan lintasan menggunakan pencahayaan ruangan, dan variasi gelap dengan pencahayaan lintasan menggunakan pencahayaan lampu senter dengan keterangan 50 lumens. Setiap variasi pengujian akan dilakukan pengujian sebanyak 50 kali.

Tabel 4. 2 Nilai kecepatan Jetbot pada satuan m/s

Tingkat Kecepatan Jetbot	Kecepatan (m/s)
0,3	0,16
0,4	0,24
0,6	0,35
0,8	0,46

4.2 Uji pendeteksian lampu lalu lintas untuk Jetbot dengan model menggunakan dataset train 500 gambar Bosch Small Traffic Light Dataset

Pengujian Jetbot dengan model yang memakai data train berupa 500 gambar dari Bosch Small Traffic Light Dataset. Gambar pada Bosch Small Traffic Light Dataset merupakan gambar lampu lalu lintas yang diambil dari keadaan asli di jalan. Pada pengujian bagian ini dibandingkan bagaimana model dengan data training 500 Bosch Small Traffic Light Dataset yang di-augmentasi dan tidak. Augmentasi data merupakan cara untuk membuat dataset mempunyai variasi bentuk yang berbeda, khususnya gambar akan diedit untuk dijadikan data training model agar model *deep learning* dapat mencari pola dari fitur lebih baik. Beberapa contoh gambar yang di-augmentasi dapat dilihat pada gambar berikut.



Gambar 4. 3 Gambar dari dataset Bosch Small Traffic Light Dataset sebelum dan sesudah di-augmentasi.

4.2.1 Uji pendeteksian lampu lalu lintas menggunakan model dengan data training 500 data Bosch

Pengujian pendeteksian lampu lalu lintas Jetbot menggunakan model ResNet18 yang di-train dengan data training yang berupa 500 data Bosch Small Traffic Light Dataset tanpa augmentasi. Pengujian dilakukan pada dua kondisi penerangan yaitu kondisi gelap dan terang. Hasil dari pengujian pendeteksian lampu lalu lintas sebagai berikut:

Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	42	40	40	38
	Gagal	8	10	10	12
Kuning	Sukses	20	14	7	4
	Gagal	30	36	43	46
Merah	Sukses	14	7	5	2
	Gagal	36	43	45	48
Akurasi		50.67%	40.67%	34.67%	29.33%

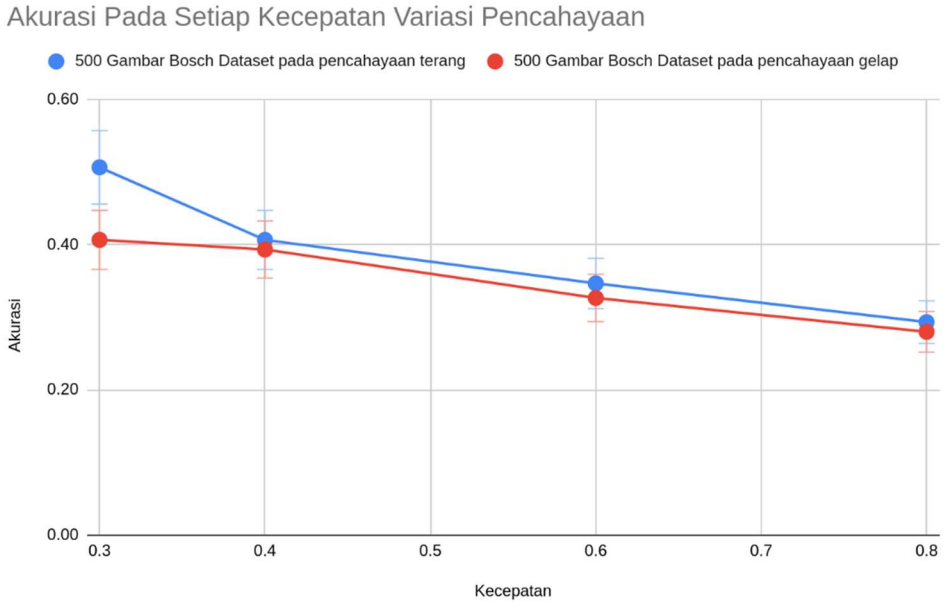
Tabel 4. 3 Pengujian dengan model yang di-train dengan 500 Gambar Bosch Small Traffic Light Dataset pada pencahayaan terang

Tabel 4. 4 Pengujian dengan model yang di-train dengan 500 Gambar Bosch Small Traffic Light Dataset pada pencahayaan gelap

	Kecepatan				
Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	40	42	39	38
	Gagal	10	8	11	12
Kuning	Sukses	13	12	7	3
	Gagal	37	38	43	47
Merah	Sukses	8	5	3	1
	Gagal	42	45	47	49
Akurasi		40.67%	39.33%	32.67%	28.00%

Dari tabel hasil pengujian Jetbot untuk mendeteksi lampu lalu lintas menggunakan model yang di-train dengan dataset train berupa 500 gambar dari Bosch Small Traffic Light Dataset pada keadaan penerangan terang dan gelap didapatkan bahwa seiring bertambahnya kecepatan Jetbot

akurasi pendeteksian mengalami penurunan. Hasil akurasi pengujian tertinggi terdapat pada kecepatan 0.3 pada kondisi terang dengan akurasi 50,67% dan hasil akurasi pengujian terendah terdapat pada kecepatan 0.8 pada kondisi gelap dengan akurasi 28,00%. Performa Jetbot dalam mendeteksi lampu lalu lintas pada kedua kondisi penerangan di tiap kecepatan dapat dilihat pada grafik berikut.



Gambar 4. 4 Grafik Akurasi di setiap kecepatan dengan variasi pencahayaan

Pada gambar 4.3 dapat dilihat penurunan nilai akurasi dari pengujian Jetbot dengan model dari data training 500 gambar Bosch Small Traffic Light Dataset seiring naiknya kecepatan. Hal ini terjadi akibat bagaimana dibutuhkan waktu untuk proses mengklasifikasi warna lampu lalu lintas dan semakin cepat Jetbot maka gambar akan semakin kurang jelas sehingga nilai akurasi saat pengujian menurun. Kebutuhan waktu dalam memproses gambar yang ditangkap oleh kamera dan mengeluarkan hasil klasifikasi yang dijalankan oleh model *neural network* membuat Jetbot sulit merespon dengan tepat pada kecepatan yang lebih tinggi. Akurasi model terbaik adalah pada kecepatan rendah yaitu tingkat 0,3, dengan akurasi 50,67% pada pencahayaan terang dan 40,67% pada pencahayaan gelap.

Pada pengujian ini dapat dilihat juga bagaimana presisi Jetbot dalam mendeteksi setiap warna lampu lalu lintas. Nilai presisi dalam Jetbot mengklasifikasi setiap warna lampu lalu lintas dapat dilihat dari tabel berikut untuk kedua penerangan.

Tabel 4. 5 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar Bosch Small Traffic Light Dataset pada kondisi terang

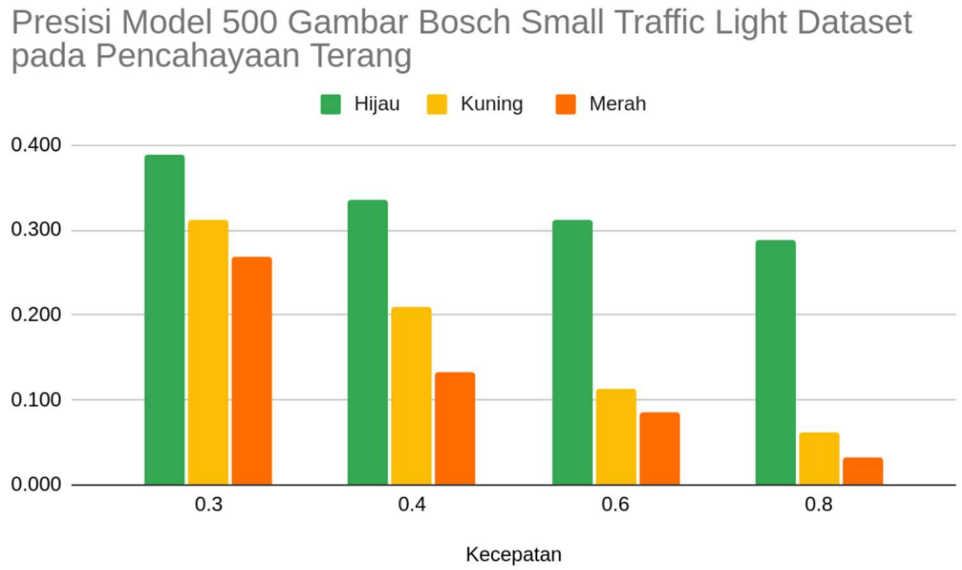
Kecepatan	Hijau	Kuning	Merah
0.3	0.389	0.313	0.269
0.4	0.336	0.209	0.132

0.6	0.313	0.113	0.086
0.8	0.288	0.063	0.033

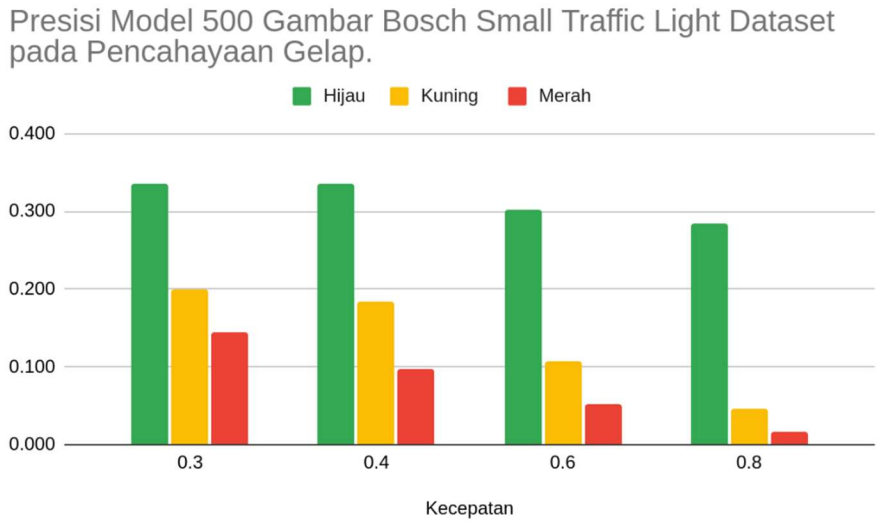
Tabel 4. 6 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar Bosch Small Traffic Light Dataset pada kondisi gelap.

Kecepatan	Hijau	Kuning	Merah
0.3	0.336	0.200	0.145
0.4	0.336	0.185	0.098
0.6	0.302	0.108	0.053
0.8	0.284	0.047	0.017

Dari kedua tabel 4.5, dan tabel 4.6 dapat dilihat bahwa nilai presisi untuk pencahayaan terang cenderung lebih baik untuk semua tingkat kecepatan, lalu nilai presisi terbaik untuk kedua kondisi pencahayaan dimiliki oleh klasifikasi warna hijau lampu lalu lintas. presisi terbaik dimiliki oleh warna hijau pada keadaan terang dengan kecepatan 0,3 sebesar 0,336, sedangkan nilai presisi paling rendah adalah untuk warna merah dengan kecepatan 0.8 yaitu sebesar 0.017 pada saat keadaan pencahayaan gelap.. Perbandingan nilai presisi secara keseluruhan dapat dilihat pada grafik berikut.



Gambar 4. 5 Grafik Presisi setiap warna untuk model dengan dataset Bosch Small Traffic Light Dataset pada kondisi terang.



Gambar 4. 6 Grafik Presisi setiap warna untuk model dengan dataset Bosch Small Traffic Light Dataset pada kondisi gelap.

Pada grafik presisi setiap warna di kedua pencahayaan gelap dan terang dapat dilihat bahwa warna hijau mempunyai nilai presisi terbaik dibanding warna kuning dan merah di semua tingkat kecepatan Jetbot. Dari grafik ini dapat dikatakan bahwa model *neural network* Resnet18 dapat mempelajari pola gambar lampu lalu lintas berwarna hijau lebih baik dibanding lampu lalu lintas berwarna merah dan kuning. Hal ini diakibatkan bahwa data *training* Bosch Small Traffic Light Dataset mempunyai warna lebih jelas dan kontras terhadap lingkungan sekitarnya dan pada pengujian kamera Jetbot menangkap warna hijau dengan baik disbanding warna kuning dan merah. Pada Bosch Small Traffic Light Dataset lampu lalu lintas berwarna kuning dan merah juga memiliki warna yang mirip yang diakibatkan oleh kamera yang digunakan pembuat data Bosch Small Traffic Light Dataset serta keadaan lingkungan sekitarnya seperti pencahayaan yang tidak selalu sama.

4.2.2 Uji pendeteksian lampu lalu lintas menggunakan model dengan data training 500 data Bosch dengan augmentasi

Pengujian pendeteksian lampu lalu lintas Jetbot menggunakan model ResNet18 yang di-train dengan data training yang berupa 500 data Bosch Small Traffic Light Dataset dengan augmentasi. Pengujian dilakukan pada dua kondisi penerangan yaitu kondisi gelap dan terang. Hasil dari pengujian pendeteksian lampu lalu lintas sebagai berikut:

Tabel 4. 7 Pengujian dengan model yang di-train dengan 500 Gambar Bosch Small Traffic Light Dataset dan augmentasi pada pencahayaan terang

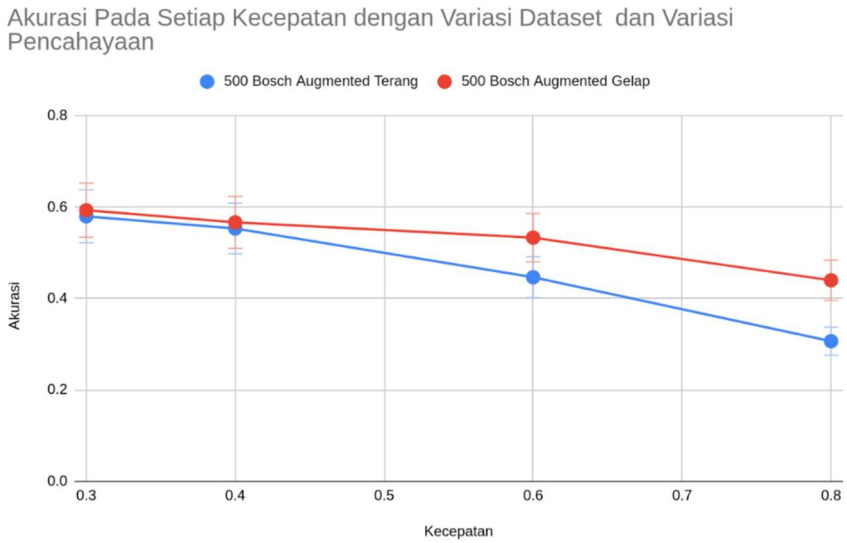
Warna Lampu Lalu Lintas	Kecepatan

		0.3	0.4	0.6	0.8
Hijau	Sukses	45	44	40	37
	Gagal	5	6	10	13
Kuning	Sukses	22	18	12	5
	Gagal	28	32	38	45
Merah	Sukses	20	21	15	4
	Gagal	30	29	35	46
Akurasi		58.00%	55.33%	44.67%	30.67%

Tabel 4. 8 Pengujian dengan model yang di-train dengan 500 Gambar Bosch Small Traffic Light Dataset dan augmentasi pada pencahayaan gelap

	Kecepatan				
Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	44	45	43	40
	Gagal	6	5	7	10
Kuning	Sukses	27	25	27	19
	Gagal	23	25	23	31
Merah	Sukses	18	15	10	7
	Gagal	32	35	40	43
Akurasi		59.33%	56.67%	53.33%	44.00%

Dari tabel hasil pengujian Jetbot untuk mendeteksi lampu lalu lintas menggunakan model yang di-train dengan dataset train berupa 500 gambar dari Bosch Small Traffic Light Dataset dengan augmentasi pada keadaan penerangan terang dan gelap didapatkan bahwa seiring bertambahnya kecepatan Jetbot akurasi pendeteksian mengalami penurunan. Hasil akurasi pengujian tertinggi terdapat pada kecepatan 0.3 pada kondisi gelap dengan akurasi 59,33% dan hasil akurasi pengujian terendah terdapat pada kecepatan 0.8 pada kondisi terang dengan akurasi 30,67%. Performa Jetbot dalam mendeteksi lampu lalu lintas pada kedua kondisi penerangan di tiap kecepatan dapat dilihat pada grafik berikut.



Gambar 4. 7 Grafik Akurasi di setiap kecepatan dengan variasi pencahayaan

Pada gambar 4.7 dapat dilihat penurunan nilai akurasi dari pengujian Jetbot dengan model dari data training 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi data seiring naiknya kecepatan. Hal ini terjadi akibat bagaimana dibutuhkan waktu untuk mengklasifikasi warna lampu lalu lintas dan semakin cepat Jetbot maka gambar akan semakin kurang jelas sehingga nilai akurasi saat pengujian menurun. Hal ini berpengaruh pada model *deep learning* menerima input yang kurang baik sehingga hasil probabilitas yang dikeluarkan oleh model *deep learning* kurang baik sehingga klasifikasi lampu lalu lintas menjadi tidak tepat.

Pada pengujian ini dapat dilihat juga bagaimana presisi Jetbot dalam mendeteksi setiap warna lampu lalu lintas. Nilai presisi dalam Jetbot mengklasifikasi setiap warna lampu lalu lintas dapat dilihat dari tabel berikut untuk kedua penerangan.

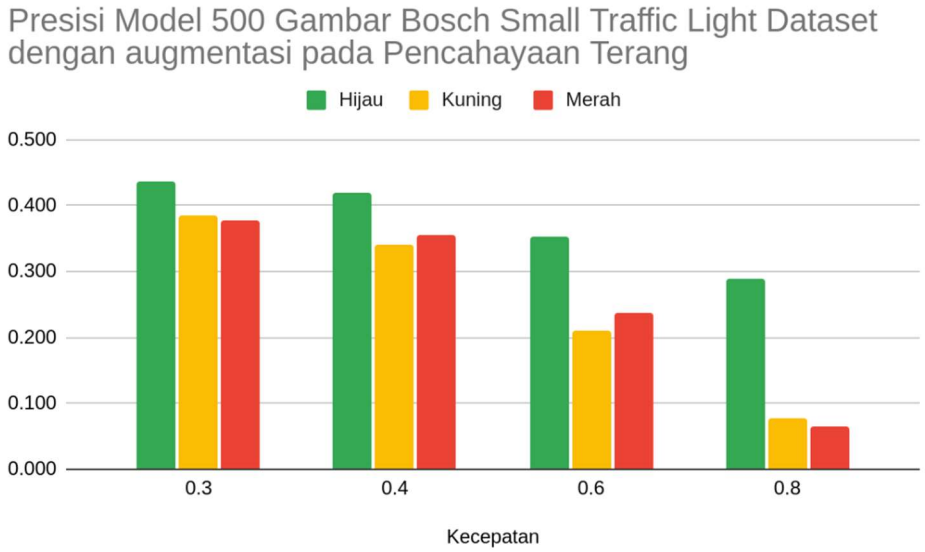
Tabel 4. 9 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi pada kondisi terang.

Kecepatan	Hijau	Kuning	Merah
0.3	0.437	0.386	0.377
0.4	0.419	0.340	0.356
0.6	0.354	0.211	0.238
0.8	0.289	0.078	0.065

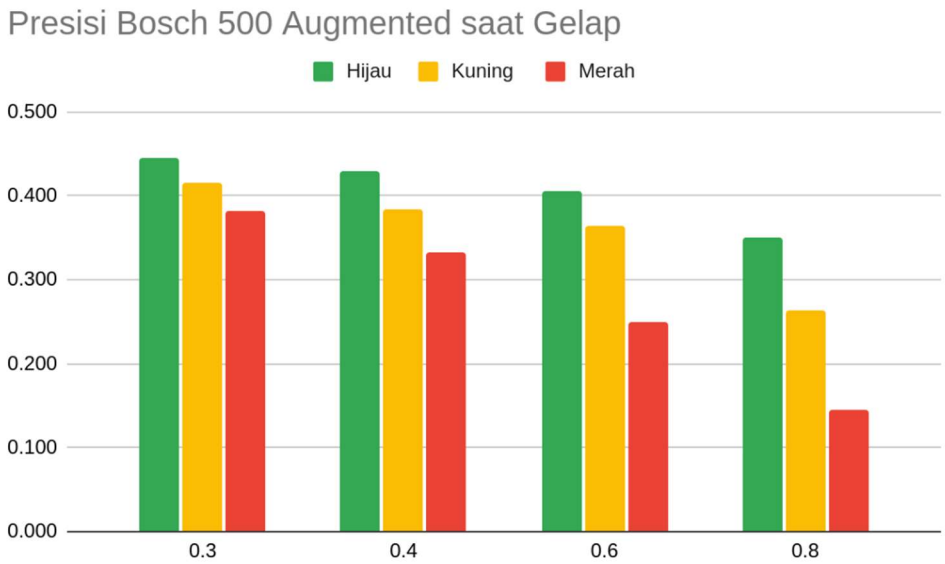
Tabel 4. 10 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi pada kondisi terang.

Kecepatan	Hijau	Kuning	Merah
0.3	0.444	0.415	0.383
0.4	0.429	0.385	0.333
0.6	0.406	0.365	0.250
0.8	0.351	0.264	0.146

Dari kedua tabel 4.9, dan tabel 4.10 dapat dilihat bahwa nilai presisi untuk pencahayaan gelap dibanding nilai presisi untuk pencahayaan terang cenderung lebih baik untuk semua tingkat kecepatan, lalu nilai presisi terbaik untuk kedua kondisi pencahayaan dimiliki oleh klasifikasi warna hijau lampu lalu lintas. Nilai presisi untuk warna hijau terbaik adalah 0,444 pada saat keadaan gelap dan dengan kecepatan 0,3, dan nilai presisi paling rendah adalah untuk warna merah pada pencahayaan terang dengan kecepatan 0.8 yaitu sebesar 0.065. Perbandingan nilai presisi secara keseluruhan dapat dilihat pada grafik berikut.



Gambar 4. 8 Grafik Presisi setiap warna untuk model dengan dataset Bosch Small Traffic Light Dataset dengan augmentasi pada kondisi terang.



Gambar 4. 9 Grafik Presisi setiap warna untuk model dengan dataset Bosch Small Traffic Light Dataset dengan augmentasi pada kondisi gelap.

Pada gambar 4.8 dan 4.9 grafik presisi setiap warna di kedua pencahayaan gelap dan terang dapat dilihat bahwa warna hijau mempunyai nilai presisi terbaik dibanding warna kuning dan merah di semua tingkat kecepatan Jetbot. Dari grafik ini dapat dikatakan bahwa model neural network Resnet18 dapat mempelajari pola gambar lampu lalu lintas berwarna hijau lebih baik dibanding lampu lalu lintas berwarna merah dan kuning. Khususnya penggunaan data training Bosch Small Traffic Light Dataset yang mempunyai karakter gambar yang berbeda dengan lingkungan pengujian penelitian ini. Warna kuning dan merah mempunyai nilai presisi lebih rendah diakibatkan oleh gambar pada dataset Bosch Small Traffic Light Dataset khususnya warna kuning dan merah di beberapa gambar cenderung mirip diakibatkan kondisi pencahayaan pada lingkungan pengambilan gambar yang tidak konsisten sehingga mempengaruhi warna pada gambar. Nilai presisi cenderung menurun untuk ketiga warna lampu lalu lintas dengan naiknya kecepatan, hal ini diakibatkan proses klasifikasi di saat kecepatan yang lebih tinggi lebih sulit karena adanya blur. Namun pada pengujian model pada Jetbot yang menggunakan data training yang sudah di-augmentasi didapatkan bahwa pada pencahayaan lintasan gelap nilai presisi untuk semua warna lebih baik dibandingkan pencahayaan terang. Hal ini terjadi karena saat pencahayaan lintasan gelap warna dari lampu lebih terang dan jelas dibanding lingkungan belakangnya sehingga model deep learning dapat lebih mengekstraksi fitur sebagai input untuk proses klasifikasi.

4.3 Uji pendeteksian lampu lalu lintas untuk Jetbot dengan model menggunakan dataset diambil dari Jetbot

Uji pendeteksian lampu lalu lintas menggunakan model yang di-*train* pada data berupa gambar yang diambil dari kamera Jetbot dibagi menjadi 2 variasi model yaitu data dengan data *training* berjumlah 250 gambar, dan data *training* berjumlah 500 gambar. Data gambar yang diambil dari kamera Jetbot dapat dilihat sebagai berikut.

4.3.1 Uji pendeteksian lampu lalu lintas menggunakan model dengan data training 250 gambar diambil dari Jetbot

Pengujian pendeteksian lampu lalu lintas Jetbot menggunakan model ResNet18 yang di-train dengan data training yang berupa 250 data gambar diambil dari Jetbot. Pengujian dilakukan pada dua kondisi penerangan yaitu kondisi gelap dan terang. Hasil dari pengujian pendeteksian lampu lalu lintas sebagai berikut:

Tabel 4. 11 Pengujian dengan model yang di-train dengan 250 gambar diambil dari Jetbot pada pencahayaan terang.

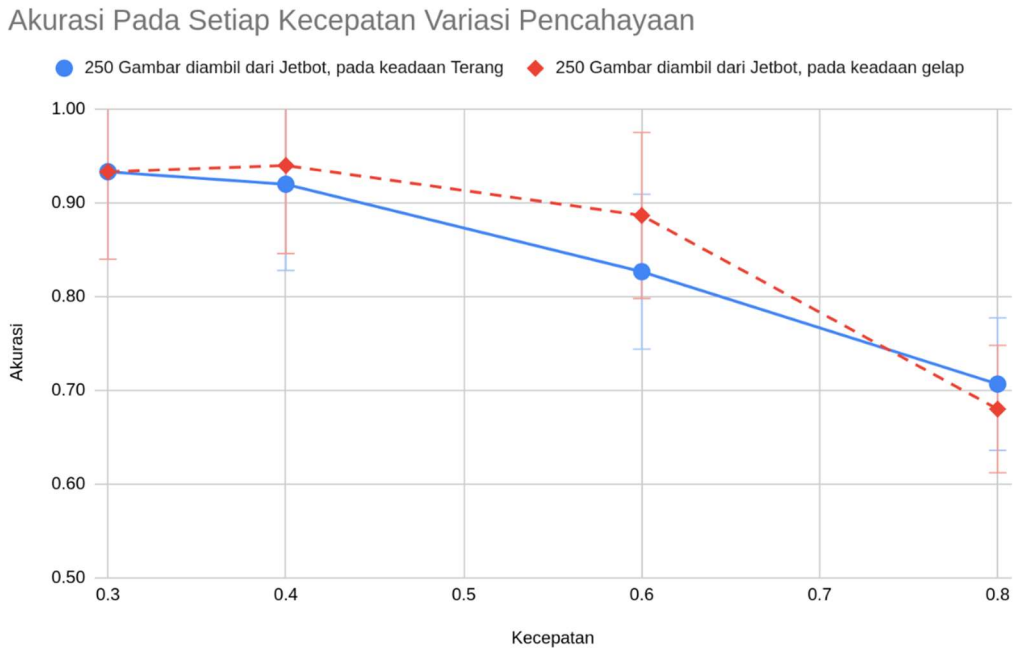
	Kecepatan				
Warna Lampu Lalu Lintas	Respon	0.3	0.4	0.6	0.8
Hijau	Sukses	50	48	49	50
	Gagal	0	2	1	0
Kuning	Sukses	48	47	40	30
	Gagal	2	3	10	20
Merah	Sukses	42	43	35	26
	Gagal	8	7	15	24
Akurasi		93.33%	92.00%	82.67%	70.67%

Tabel 4. 12 Pengujian dengan model yang di-train dengan 250 gambar diambil dari Jetbot pada pencahayaan gelap.

	Kecepatan				
Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	50	50	49	44
	Gagal	0	0	1	6
Kuning	Sukses	48	48	44	38
	Gagal	2	2	6	12
Merah	Sukses	42	43	40	20
	Gagal	8	7	10	30
Akurasi		93.33%	94.00%	88.67%	68.00%

Dari tabel hasil pengujian Jetbot untuk mendeteksi lampu lalu lintas menggunakan model yang di-*train* dengan *dataset training* berupa 250 gambar diambil dari kamera Jetbot, pada keadaan penerangan terang dan gelap didapatkan bahwa seiring bertambahnya kecepatan Jetbot, akurasi pendeteksian mengalami penurunan. Hasil akurasi pengujian tertinggi terdapat pada kecepatan 0,4 pada kondisi penerangan gelap dengan akurasi 94% dan hasil akurasi pengujian terendah terdapat

pada kecepatan 0,8 pada kondisi gelap dengan akurasi 68%. Performa Jetbot dalam mendeteksi lampu lalu lintas pada kedua kondisi penerangan di tiap kecepatan dapat dilihat pada grafik berikut.



Gambar 4. 10 Grafik Akurasi di setiap kecepatan dengan variasi pencahayaan

Pada gambar 4.10 dapat dilihat penurunan nilai akurasi dari pengujian Jetbot dengan model dari data *training* 250 gambar diambil dari Jetbot seiring naiknya kecepatan. Hal ini terjadi akibat bagaimana dibutuhkan waktu untuk mengklasifikasi warna lampu lalu lintas dan semakin cepat Jetbot maka gambar akan semakin kurang jelas seperti blur, sehingga nilai akurasi saat pengujian menurun.

Pada pengujian ini dapat dilihat juga bagaimana presisi Jetbot dalam mendeteksi setiap warna lampu lalu lintas. Nilai presisi dalam Jetbot mengklasifikasi setiap warna lampu lalu lintas dapat dilihat dari tabel berikut untuk kedua penerangan.

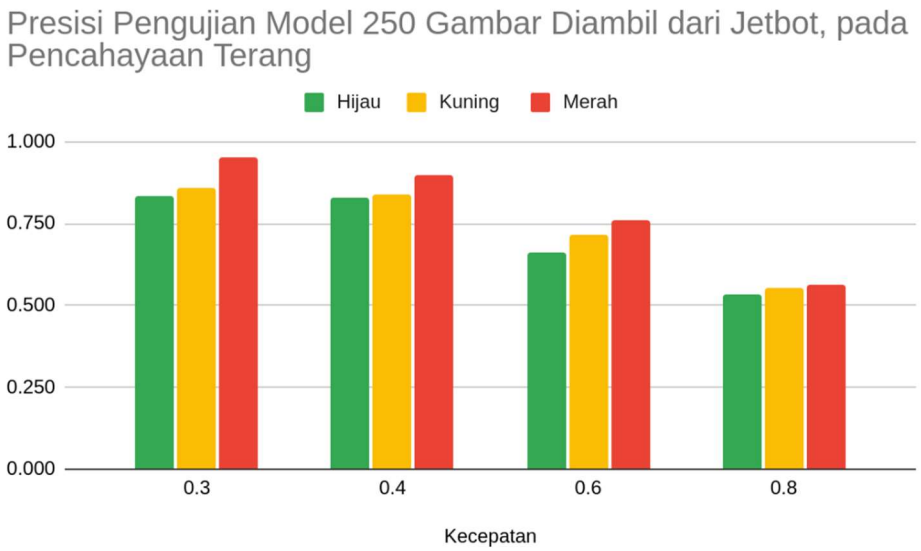
Tabel 4. 13 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 250 gambar diambil dari Jetbot, pada penerangan terang.

Kecepatan	Hijau	Kuning	Merah
0.3	0.833	0.857	0.955
0.4	0.828	0.839	0.896
0.6	0.662	0.714	0.761
0.8	0.532	0.556	0.565

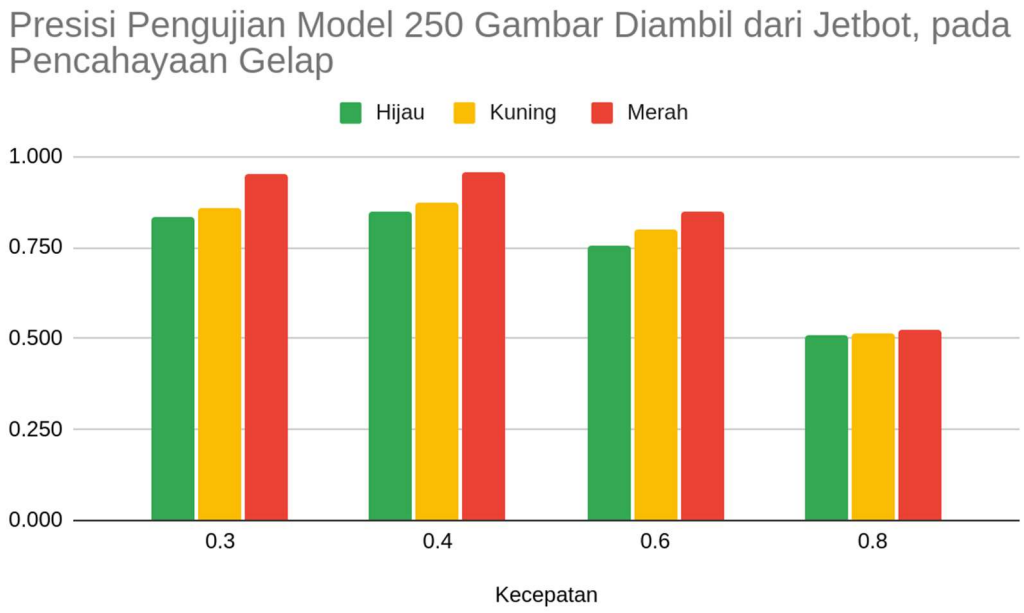
Tabel 4. 14 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 250 gambar diambil dari Jetbot, pada penerangan gelap.

Kecepatan	Hijau	Kuning	Merah
0.3	0.833	0.857	0.955
0.4	0.847	0.873	0.956
0.6	0.754	0.800	0.851
0.8	0.512	0.514	0.526

Dari kedua tabel 4.13, dan tabel 4.14 dapat dilihat bahwa nilai presisi setiap warna pada pencahayaan gelap cenderung lebih baik untuk semua tingkat kecepatan, lalu nilai presisi terbaik untuk kedua kondisi pencahayaan dimiliki oleh klasifikasi warna merah lampu lalu lintas. Nilai presisi untuk warna merah terbaik adalah pada saat keadaan penerangan gelap di kecepatan 0,4, dan nilai presisi paling rendah adalah untuk warna hijau adalah pada kecepatan 0,8, pada penerangan gelap yaitu sebesar 0,512. Perbandingan nilai presisi secara keseluruhan dapat dilihat pada grafik berikut.



Gambar 4. 11 Grafik presisi pengujian setiap warna untuk model dengan dataset 250 gambar dari Jetbot, pada kondisi terang.



Gambar 4. 12 Grafik presisi pengujian setiap warna untuk model dengan dataset 250 gambar dari Jetbot, pada kondisi gelap.

Pada gambar 4.11 dan 4.12 grafik presisi setiap warna di kedua pencahayaan gelap dan terang dapat dilihat bahwa warna hijau mempunyai nilai presisi terbaik dibanding warna kuning dan merah di semua tingkat kecepatan Jetbot. Dari grafik ini dapat dikatakan bahwa model *neural network* Resnet18 dapat mempelajari pola gambar lampu lalu lintas berwarna merah lebih baik dibanding lampu lalu lintas berwarna hijau dan kuning. Model *deep learning* pada bagian ini menggunakan data training yang diambil dengan kamera pada Jetbot, hal ini mengakibatkan nilai presisi yang tinggi karena data *training* yang digunakan sudah mempunyai kondisi gambar yang sama dengan kondisi saat eksperimen. Nilai presisi cenderung menurun untuk ketiga warna lampu lalu lintas dengan naiknya kecepatan, hal ini diakibatkan proses klasifikasi di saat kecepatan yang lebih tinggi lebih sulit karena adanya blur, serta waktu proses yang dibutuhkan lebih cepat agar Jetbot dapat merespon cenderung menurun untuk ketiga warna lampu lalu lintas dengan naiknya kecepatan. Namun pada pengujian model pada Jetbot yang menggunakan data 250 gambar data yang diambil dari kamera Jetbot didapatkan bahwa pada pencahayaan lintasan gelap nilai presisi untuk semua warna lebih baik dibandingkan pencahayaan terang. Hal ini terjadi karena saat pencahayaan lintasan gelap warna dari lampu lebih terang dan jelas dibanding lingkungan belakangnya sehingga model *deep learning* dapat lebih mengekstraksi fitur sebagai input untuk proses klasifikasi.

4.3.2 Uji pendeteksian lampu lalu lintas menggunakan model dengan data training 500 gambar diambil dari Jetbot

Pengujian pendeteksian lampu lalu lintas Jetbot menggunakan model ResNet18 yang di-train dengan data training yang berupa 500 data gambar diambil dari Jetbot. Pengujian dilakukan pada dua kondisi penerangan yaitu kondisi gelap dan terang. Hasil dari pengujian pendeteksian lampu lalu lintas sebagai berikut:

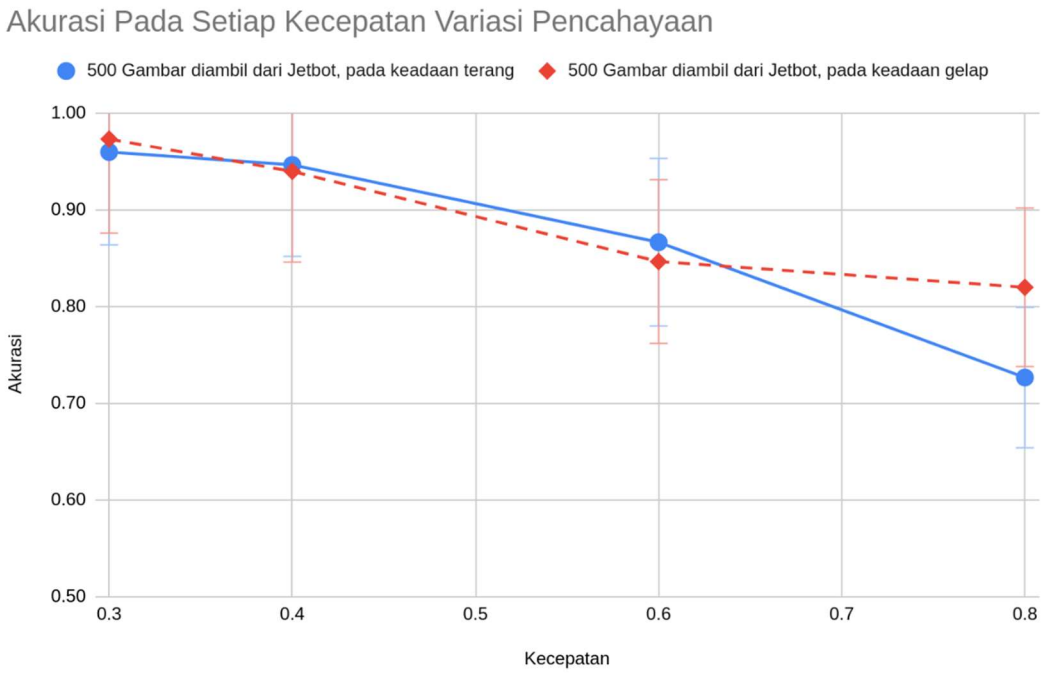
Tabel 4. 15 Pengujian dengan model yang di-train dengan 500 gambar diambil dari Jetbot pada pencahayaan terang.

		Kecepatan			
Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	50	50	48	47
	Gagal	0	0	2	3
Kuning	Sukses	47	46	42	30
	Gagal	3	4	8	20
Merah	Sukses	47	46	40	32
	Gagal	3	4	10	18
Akurasi		96.00%	94.67%	86.67%	72.67%

Tabel 4. 16 Pengujian dengan model yang di-train dengan 500 gambar diambil dari Jetbot pada pencahayaan gelap.

		Kecepatan			
Warna Lampu Lalu Lintas		0.3	0.4	0.6	0.8
Hijau	Sukses	50	48	49	43
	Gagal	0	2	1	7
Kuning	Sukses	47	47	38	40
	Gagal	3	3	12	10
Merah	Sukses	49	46	40	40
	Gagal	1	4	10	10
Akurasi		97.33%	94.00%	84.67%	82.00%

Dari tabel hasil pengujian Jetbot untuk mendeteksi lampu lalu lintas menggunakan model yang di-*train* dengan dataset *train* berupa 500 gambar diambil dari Jetbot, pada keadaan penerangan terang dan gelap didapatkan bahwa seiring bertambahnya kecepatan Jetbot akurasi pendeteksian mengalami penurunan. Hasil akurasi pengujian tertinggi terdapat pada kecepatan 0,3 pada kondisi penerangan gelap dengan akurasi 97,33% dan hasil akurasi pengujian terendah terdapat pada kecepatan 0,8 pada kondisi penerangan terang dengan akurasi 72,67%. Performa Jetbot dalam mendeteksi lampu lalu lintas pada kedua kondisi penerangan di tiap kecepatan dapat dilihat pada grafik berikut.



Gambar 4. 13 Grafik Akurasi di setiap kecepatan dengan variasi pencahayaan

Pada gambar 4.13 dapat dilihat penurunan nilai akurasi dari pengujian Jetbot dengan model dari data training 500 gambar diambil dari Jetbot seiring naiknya kecepatan. Hal ini terjadi akibat bagaimana dibutuhkan waktu untuk mengklasifikasi warna lampu lalu lintas dan semakin cepat Jetbot maka gambar akan semakin kurang jelas sehingga nilai akurasi saat pengujian menurun. Pada tingkat kecepatan 0.6 dengan nilai 0.35 m/s dibutuhkan waktu proses yang cukup cepat dan mempunyai hasil dengan probabilitas yang tinggi untuk setiap warna lampu lalu lintas, namun gambar yang ditangkap oleh kamera saat Jetbot berjalan dengan kecepatan 0.6 menjadi kurang baik, sehingga penurunan tingkat akurasi menurun.

Pada pengujian ini dapat dilihat juga bagaimana presisi Jetbot dalam mendeteksi setiap warna lampu lalu lintas. Nilai presisi dalam Jetbot mengklasifikasi setiap warna lampu lalu lintas dapat dilihat dari tabel berikut untuk kedua penerangan.

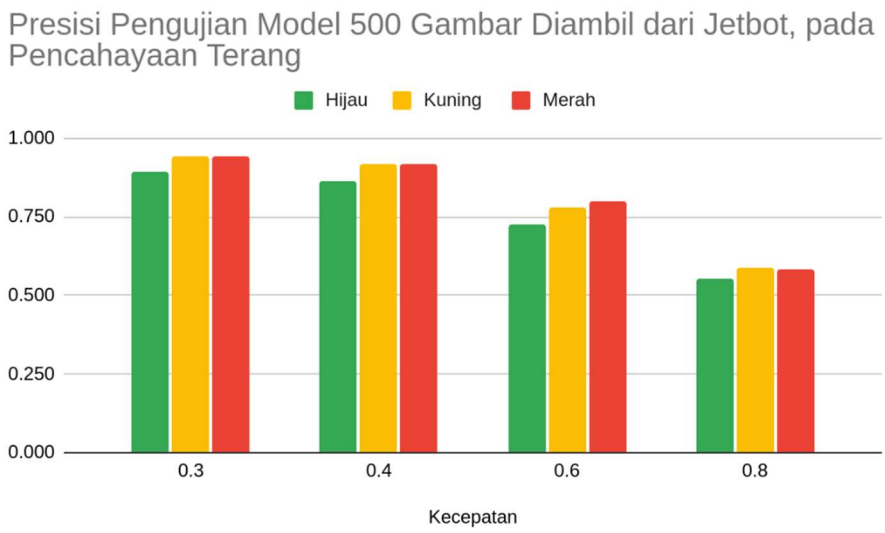
Tabel 4. 17 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar diambil dari Jetbot, pada penerangan terang.

Kecepatan	Hijau	Kuning	Merah
0.3	0.893	0.940	0.940
0.4	0.862	0.920	0.920
0.6	0.727	0.778	0.800
0.8	0.553	0.588	0.582

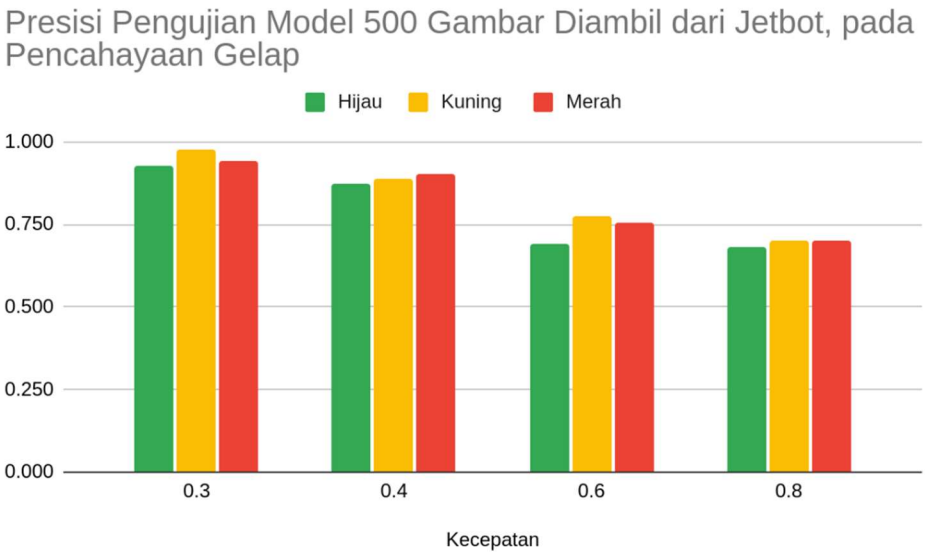
Tabel 4. 18 Nilai Presisi klasifikasi warna lampu lalu lintas Jetbot dengan model dengan data train 500 gambar diambil dari Jetbot, pada penerangan gelap.

Kecepatan	Hijau	Kuning	Merah
0.3	0.926	0.979	0.942
0.4	0.873	0.887	0.902
0.6	0.690	0.776	0.755
0.8	0.683	0.702	0.702

Dari kedua tabel 4.17, dan tabel 4.18 dapat dilihat bahwa nilai presisi menurun dengan meningkatnya kecepatan, lalu nilai presisi terbaik untuk kedua kondisi pencahayaan dimiliki oleh klasifikasi warna kuning lampu lalu lintas. Nilai presisi untuk warna terbaik adalah warna kuning pada saat keadaan gelap dan dengan kecepatan 0,3, dan nilai presisi paling rendah adalah untuk warna hijau dengan kecepatan 0.8 pada penerangan gelap yaitu sebesar 0.553. Perbandingan nilai presisi secara keseluruhan dapat dilihat pada grafik berikut.



Gambar 4. 14 Grafik presisi pengujian setiap warna untuk model dengan dataset 500 gambar dari Jetbot, pada kondisi terang.

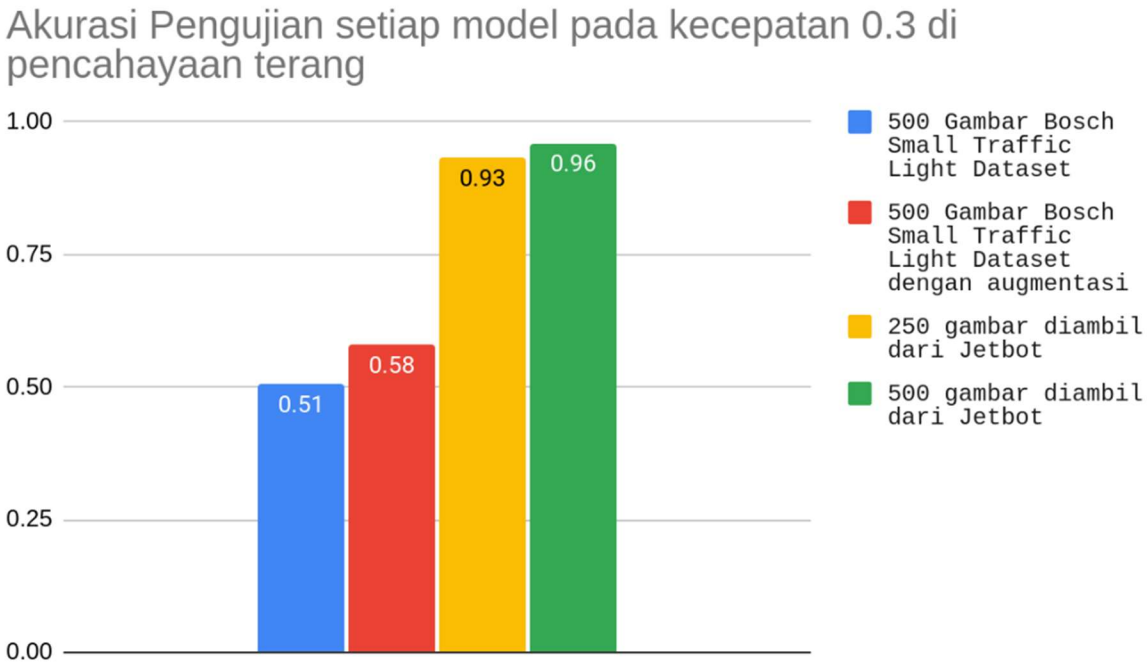


Gambar 4. 15 Grafik presisi pengujian setiap warna untuk model dengan dataset 500 gambar dari Jetbot, pada kondisi gelap.

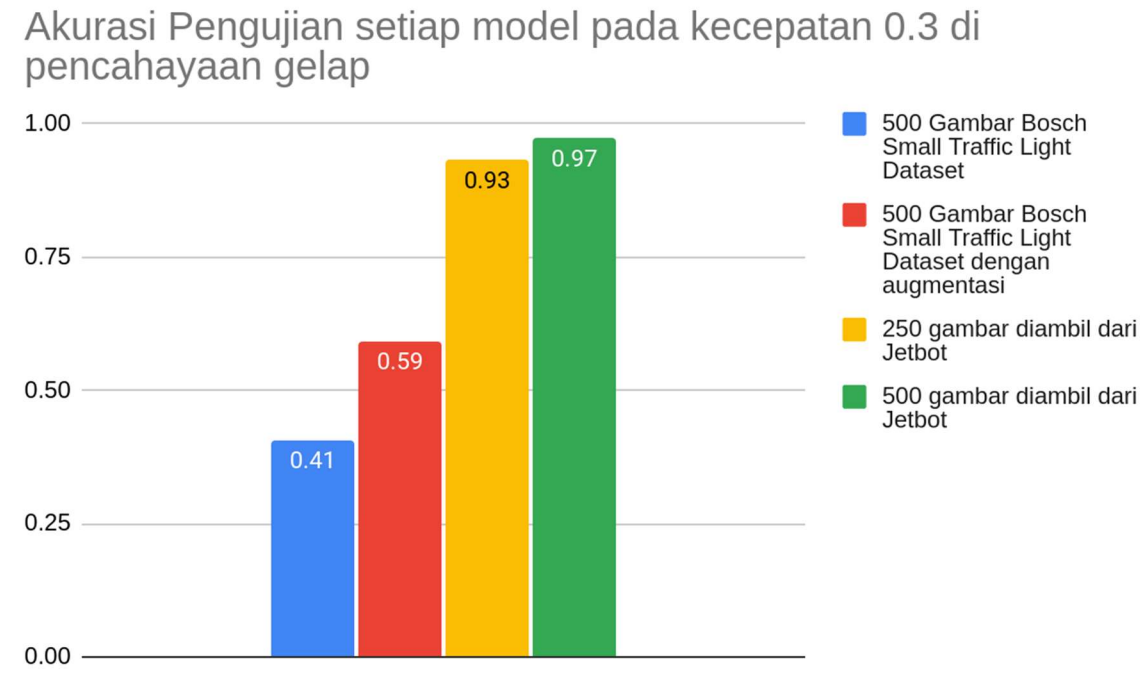
Pada gambar 4.14 dan 4.15 grafik presisi setiap warna di kedua pencahayaan gelap dan terang dapat dilihat bahwa warna hijau mempunyai nilai presisi terbaik dibanding warna kuning dan merah di semua tingkat kecepatan Jetbot. Dari grafik ini dapat dikatakan bahwa model *neural network* Resnet18 dapat mempelajari pola gambar lampu lalu lintas berwarna merah lebih baik dibanding lampu lalu lintas berwarna hijau dan kuning. Model *deep learning* pada bagian ini menggunakan data training yang diambil dengan kamera pada Jetbot, hal ini mengakibatkan nilai presisi yang tinggi karena data training yang digunakan sudah mempunyai kondisi gambar yang sama dengan kondisi saat eksperimen. Nilai presisi cenderung menurun untuk ketiga warna lampu lalu lintas dengan naiknya kecepatan, hal ini diakibatkan proses klasifikasi di saat kecepatan yang lebih tinggi lebih sulit karena adanya *blur*, serta waktu proses yang dibutuhkan lebih cepat agar Jetbot dapat merespon dengan baik. Pengujian model pada Jetbot yang menggunakan data 500 gambar data yang diambil dari kamera Jetbot didapatkan bahwa pada pencahayaan lintasan gelap nilai presisi untuk semua warna lebih baik dibandingkan pencahayaan terang. Hal ini terjadi karena saat pencahayaan lintasan gelap warna dari lampu lebih terang dan jelas dibanding lingkungan belakangnya sehingga model *deep learning* dapat lebih mengekstraksi fitur sebagai input untuk proses klasifikasi.

4.4 Perbandingan performa model

Performa model untuk keseluruhan dapat dilihat dari nilai akurasi pengujian di tingkat kecepatan 0.3. Performa model akan diamati bagaimana akurasinya tiap variasi pencahayaan. Setiap model mempunyai performa yang berbeda pada keadaan pencahayaan gelap dan terang. Lalu dapat diamati juga performa secara keseluruhan untuk suatu model dalam mendeteksi di keadaan gelap dan terang, serta di kecepatan 0.3 setiap model saat pengujian mempunyai nilai akurasi terbaik dibanding di kecepatan lainnya. Nilai akurasi dari setiap model pada variasi pencahayaan terang dan gelap di kecepatan 0.3 dapat dilihat pada grafik berikut ini.



(a)



(b)

Gambar 4. 16 Grafik performa akurasi model pengujian setiap kondisi pencahayaan di kecepatan 0.3. (a) Kondisi pencahayaan terang (b) Kondisi pencahayaan gelap.

Pada grafik ini dapat dilihat bahwa nilai tertinggi didapatkan oleh model dengan data training gambar diambil dari Jetbot di pengujian keadaan pencahayaan gelap. Lalu nilai terendah adalah model dengan data training Bosch Small Traffic Light Dataset dengan jumlah 500 gambar di pengujian keadaan pencahayaan gelap. Perbedaan performa antara model dengan data training gambar dari Bosch Small Traffic Light Dataset dengan jumlah 500 gambar tanpa augmentasi dan dengan augmentasi, dengan model yang menggunakan dataset training berupa gambar yang diambil dari Jetbot teramati cukup signifikan. Dapat dilihat nilai tertinggi dari model yang menggunakan dataset Bosch Small Traffic Light Dataset adalah 59,33%, dan nilai terendah oleh model yang menggunakan data training gambar dari Jetbot adalah 93,33%. Terdapat perbedaan sebesar 31,34%. Namun performa antara model yang menggunakan data training Bosch Small Traffic Light Dataset mempunyai perbedaan antara model yang menggunakan data training yang di-augmentasi dan tidak. Model dengan data training Bosch Small Traffic Light Dataset dengan augmentasi mempunyai nilai akurasi lebih tinggi dari model dengan data training Bosch Small Traffic Light Dataset tanpa augmentasi dengan nilai akurasi 58,00% pada keadaan terang dan 59,33% pada keadaan penerangan gelap. Hal ini disebabkan oleh augmentasi gambar yang membuat model *neural network* dapat membuat pola lebih baik karena augmentasi gambar membuat gambar pada dataset mempunyai bentuk-bentuk dan perspektif yang berbeda yang dapat membuat model dapat mengklasifikasi lebih baik. Nilai terbaik dari model yang menggunakan gambar yang diambil dari data Jetbot terdapat pada model yang menggunakan gambar dari Jetbot berjumlah 500 gambar dengan nilai 96,00% pada keadaan penerangan terang, dan nilai akurasi 97,33% pada keadaan penerangan gelap. Model Jetbot yang menggunakan 500 gambar yang diambil dari Jetbot merupakan model terbaik dari seluruh variasi model hal ini disebabkan oleh jumlah data yang banyak dan konten data yang mempunyai keadaan yang sama dengan lingkungan pengujian, sehingga model *neural network* dapat lebih mudah menemukan pola. Performa model dengan nilai akurasi setiap model di kecepatan 0.3 dari kedua keadaan pencahayaan dapat dilihat pada tabel berikut ini.

Tabel 4. 19 Nilai akurasi model pada pengujian kondisi penerangan terang di kecepatan 0,3.

Model	Nilai Akurasi
Model dengan 500 gambar Bosch Small Traffic Light Dataset	50.67%
Model dengan 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi	58.00%
Model dengan 250 gambar dari Jetbot	93.33%
Model dengan 500 gambar dari Jetbot	96.00%

Tabel 4. 20 Nilai akurasi model pada pengujian kondisi penerangan gelap di kecepatan 0,3.

Model	Nilai Akurasi
Model dengan 500 gambar Bosch Small Traffic Light Dataset	40.67%
Model dengan 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi	59.33%
Model dengan 250 gambar dari Jetbot	93.33%
Model dengan 500 gambar dari Jetbot	97.33%

Dari tabel 4.19 nilai tertinggi akurasi model terbaik di kecepatan 0.3 pada keadaan terang adalah model dengan 500 gambar dari Jetbot dengan nilai 96%, dan terendah adalah model dengan 500 gambar dari Bosch Small Traffic Light Dataset dengan nilai 50,67%. Dari tabel 4.20 nilai tertinggi akurasi model terbaik di kecepatan 0.3 pada keadaan terang adalah model dengan 500 gambar dari Jetbot dengan nilai 97,33%, dan terendah adalah model dengan 500 gambar dari Bosch Small Traffic Light Dataset dengan nilai 40,67%. Namun augmentasi data dapat membuat performa model dengan data train Bosch Small Traffic Light Dataset berkembang dari 50,67% menjadi 58,00% pada keadaan pencahayaan terang dan berkembang dari 40,67% menjadi 59,33% setelah di-augmentasi. Penggunaan dataset yang merupakan data yang sudah dipotong atau di*crop* membuat kualitas gambar turun, perlunya penelitian lebih lanjut bagaimana pengaruh tentang ketajaman gambar terhadap performa model *machine learning* dalam membuat pola untuk mempelajari lampu lalu lintas serta warnanya yang dapat mempengaruhi performa. Karena sebuah model *machine learning* untuk aplikasi *image classification* kebanyakan membutuhkan resolusi *input* yang tidak terlalu besar seperti 256 pixels x 256 pixels. Sehingga ketajaman suatu gambar yang dicapai dengan resolusi tersebut tidak memungkinkan terlalu baik untuk suatu gambar. Ketajaman gambar dapat mempengaruhi performa model *machine learning* dalam klasifikasi gambar ketika fitur yang dibutuhkan hilang seperti tekstur, dan warna sampai tidak terlihat. (Dodge & Karam, 2016) Pada penelitian ini gambar-gambar *dataset* yang digunakan sebagai data *training* masih di ketajaman yang tidak kehilangan fiturnya yaitu warna dan tekstur yang dibutuhkan untuk membedakan warna lampu lalu lintas. Dapat dilakukan penelitian dengan menggunakan data yang sama-sama dicrop agar melihat pengaruh dataset gambar yang di-*crop* dengan ketajaman yang berkurang. Terkait akurasi Jetbot yang menurun dalam mengklasifikasi lampu lalu lintas seiring naiknya kecepatan diakibatkan karena waktu proses klasifikasi yang sama, hal ini dilihat ketika dilakukan percobaan dengan titik jarak perubahan warna lampu yang ditambahkan agar lebih panjang terlihat respon yang lebih baik, hipotesa ini dapat dibuktikan lagi agar terbukti untuk penelitian selanjutnya. Kecepatan proses juga dapat dipercepat dengan menggunakan model dengan arsitektur yang lebih ringan secara komputasi.

Peningkatan performa dapat dilakukan dengan beberapa cara seperti penggunaan dataset hibrid yaitu dataset *training* dengan dua tipe dataset yaitu dataset dari dataset publik yang mempunyai gambar yang tidak sama dengan kondisi pengujian namun memiliki kesamaan dengan lingkungan pengujian digabung dengan gambar yang diambil dari lingkungan pengujian. Cara kedua yaitu dengan menggunakan model yang lebih rumit agar mendapatkan akurasi lebih baik, atau dengan model dengan arsitektur yang tidak terlalu berat namun berakurasi tinggi seperti MobileNetV2. Berikut salah satu data pengujian di tingkat kecepatan Jetbot sebesar 0,3 dengan data hibrid berupa 350 gambar dari Bosch Small Traffic Light Dataset dan 150 gambar diambil dari Jetbot dengan model ResNet18 dan pengujian di tingkat kecepatan Jetbot sebesar 0,3 dengan model MobileNetV2 dengan data training berupa 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi.

Tabel 4. 21 Akurasi pengujian menggunakan data hibrid dan model MobileNetV2

Warna Lampu Lalu Lintas	350 Bosch - 150 Jetbot	MobileNetv2- 500 Augmented
	Kecepatan	

		0,3(Terang)	0,3(Gelap)	0,3(Terang)	0,3(Gelap)
Hijau	Sukses	46	48	47	48
	Gagal	5	2	3	2
Kuning	Sukses	27	30	20	27
	Gagal	28	20	30	23
Merah	Sukses	22	29	25	26
	Gagal	30	21	25	24
Akurasi		60,13%	71,33%	61,33%	67,33%

Dari table 4.21 dapat dilihat perkembangan akurasi menggunakan dataset hibrid dan menggunakan model MobileNetV2. Perkembangan nilai akurasi klasifikasi penggunaan dataset hibrid terhadap klasifikasi menggunakan model dengan data *training* 500 gambar Bosch Small Traffic Light Dataset tanpa augmentasi yaitu sebesar 9,46% di keadaan pencahayaan terang dan 19,46% di keadaan pencahayaan gelap. Perkembangan nilai akurasi klasifikasi penggunaan dataset hibrid terhadap klasifikasi menggunakan model dengan data *training* 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi gambar yaitu sebesar 2,13% di keadaan pencahayaan lintasan terang dan 12% di keadaan pencahayaan lintasan gelap. Perkembangan performa menggunakan dataset hibrid dibandingkan dengan dataset 500 gambar Bosch Small Traffic Light Dataset diakibatkan karena model *machine learning* dapat mempelajari lingkungan pengujian lebih baik karena adanya distribusi data gambar yang sama dengan lingkungan pengujiannya dengan proporsi 70% - 30% pada dataset hibrid dengan 70% menggunakan gambar dari Bosch Small Traffic Light Dataset dan 30% untuk gambar yang diambil dari Jetbot. Sehingga model *machine learning* mempunyai keyakinan lebih tinggi saat mengklasifikasikan warna lampu lalu lintas di saat pengujian. Perkembangan nilai akurasi klasifikasi penggunaan model MobileNetV2 dengan data *training* 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi gambar dibandingkan dengan model Resnet18 menggunakan data *training* 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi gambar yaitu sebesar 3,33% di keadaan pencahayaan lintasan terang dan 8% di keadaan pencahayaan lintasan gelap. Perkembangan nilai akurasi klasifikasi ini terjadi diakibatkan karena MobileNetV2 mempunyai arsitektur yang tidak terlalu dalam tetapi mempunyai struktur yang berbeda dengan ResNet18, dengan penggunaan BottleNeck layer yang merupakan *layer* pada arsitektur *neural network* yang menggunakan konsep narrow-wide-narrow untuk pengolahan parameter agar arsitektur MobileNetV2 dapat berjalan dengan ringan sehingga pemrosesan dapat dilakukan lebih cepat, dan tanpa mengurangi performa model.(Sandler et al, 2018)

“Halaman ini sengaja dikosongkan”

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilaksanakan, didapatkan beberapa poin yang dapat disimpulkan dari hasil penelitian. Berikut kesimpulan dari penelitian ini.

1. Kecepatan 0.16 m/s (tingkat kecepatan 0.3) merupakan kecepatan terbaik untuk NVIDIA Jetbot mendeteksi warna lampu lalu lintas dengan tepat. Nilai akurasi pengujian untuk setiap model dengan variasi data *training* yang berbeda akan turun dengan naiknya kecepatan NVIDIA Jetbot. Nilai akurasi pengujian NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas pada tingkat kecepatan 0,3 pada variasi dataset *training* 500 jumlah gambar Bosch Small Traffic Light Dataset adalah 50,67% pada keadaan penerangan terang, dan 40,67% pada keadaan penerangan gelap. Nilai akurasi pengujian NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas pada tingkat kecepatan 0,3 pada variasi dataset *training* 500 jumlah gambar Bosch Small Traffic Light Dataset di-augmentasi yang merupakan proses manipulasi gambar untuk membantu model mempelajari pola lebih baik adalah 58,33% pada keadaan penerangan terang, dan 59,33% pada keadaan penerangan gelap. Nilai akurasi pengujian NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas pada tingkat kecepatan 0,3 pada variasi dataset *training* 250 jumlah gambar diambil dari kamera Jetbot adalah 93,33% pada keadaan penerangan terang, dan 93,33% pada keadaan penerangan gelap. Nilai akurasi pengujian NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas pada tingkat kecepatan 0,3 pada variasi dataset *training* 500 jumlah gambar diambil dari kamera Jetbot adalah 96,00% pada keadaan penerangan terang, dan 97,33% pada keadaan penerangan gelap.
2. Pengaruh kondisi pencahayaan lintasan dengan performa NVIDIA Jetbot untuk variasi data *training* 500 gambar Bosch Small Traffic Light Dataset di-augmentasi, 250 gambar diambil dari kamera Jetbot, dan 500 gambar diambil dari kamera Jetbot adalah pada kondisi penerangan gelap membuat akurasi NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas dengan tepat lebih baik.
3. Pengaruh data *training* untuk performa NVIDIA Jetbot dalam mendeteksi warna lampu lalu lintas didapatkan bahwa data dengan gambar yang diambil dari kamera Jetbot lebih baik dibanding dengan data dengan gambar dari Bosch Small Traffic Light Dataset. Augmentasi data dapat meningkatkan performa model dengan data *training* Bosch Small Traffic Light Dataset dilihat pada akurasi pengujian di kecepatan 0,3 pada pencahayaan terang dengan meningkatnya akurasi sebesar 7,33%.
4. Pada penelitian ini ditemukan bahwa pada tingkat kecepatan 0,3 penggunaan dataset hibrid yaitu 350 gambar Bosch Small Traffic Light Dataset dengan 150 gambar diambil dari Jetbot meningkatkan performa dalam klasifikasi lampu lalu lintas. Perkembangan terhadap klasifikasi menggunakan model dengan data *training* 500 gambar Bosch Small Traffic Light Dataset dengan augmentasi gambar yaitu sebesar 2,13% di keadaan pencahayaan lintasan terang dan 12% di keadaan pencahayaan lintasan gelap. Penggunaan dataset hibrid dapat mengurangi usaha untuk membuat *dataset* baru untuk tiap aplikasi klasifikasi gambar.

5.2 Saran

Dengan selesainya dilakukan penelitian ini, diperolehbeberapa saran guna pengembangan lanjutan dari laporan ini. Adapun saran dari penelitian ini adalah sebagai berikut.

1. Menggunakan konfigurasi *closed-loop* untuk *motor controller.*
2. Menambahkan pendeteksian untuk jenis lampu lalu lintas lain seperti lampu lalu lintas dengan tanda berbelok, atau putar balik.
3. Melakukan penelitian terkait pengaruh ketajaman gambar yang digunakan sebagai data *training*.
4. Melakukan penelitian terkait proporsi optimal untuk dataset hibrid.
5. Menambahkan kendaraan lain pada pengujian lampu lalu lintas.
6. Menggunakan Robot Operating System(ROS) pada NVIDIA Jetbot agar dapat menggunakan algoritma yang lebih baik.

DAFTAR PUSTAKA

Zhang, A. (2010). Dive into Deep Learning Release 0.17.5 (3rd ed., pp. 103-117).

Aggarwal, C. C. (2018). Neural Networks and Deep Learning. Springer International Publishing. https://doi.org/10.1007/978-3-319-94463-0

Ding, B., Qian, H., & Zhou, J. (2018). Activation Functions and Their Characteristics in Deep Neural Networks. IEEE. https://doi.org/10.1109/CCDC.2018.8407425

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). Foundations of Machine Learning Second Edition. MIT Press. https://cs.nyu.edu/~mohri/mlbook/

He, K., Zhang, X., Ren, S., Sun, J.(2015). Deep Residual Learning for Image Recognition. Arxiv. https://doi.org/10.48550/arXiv.1512.03385

Girshick, R. (2015). Fast R-CNN. Microsoft. https://doi.org/10.48550/arXiv.1504.08083

Kawakura, S., & Shibasaki, R. (2020). Deep Learning-Based Self-Driving Car: JetBot with NVIDIA AI Board to Deliver Items at Agricultural Workplace with Object-Finding and Avoidance Functions. https://doi.org/10.24018/ejfood.2020.2.3.45.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once:Unified, Real-Time Object Detection. University of Washington, Allen Institute for AI, Facebook AI Research. https://doi.org/10.48550/arXiv.1506.02640

Hofesmann, E., (2020, August 25). IoU a better detection evaluation metric. https://towardsdatascience.com/iou-a-better-detection-evaluation-metric-45a511185be1

Purva91. (2020, September 4). Precision vs. Recall – An Intuitive Guide for Every Machine Learning Person. https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/

Saha, S.,(2018, December 16). A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Agarwal, R.(2019, September 18). The 5 Classification Evaluation metrics every Data Scientist must know. https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226

Ganesh, S.(2020, July 25). What's The Role Of Weights And Bias In a Neural Network?. https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f

Rezkia, S. M.(2021, June 7). Memahami Perbedaan Algoritma Machine Learning vs Deep Learning. https://www.dqlab.id/memahami-perbedaan-algoritma-machine-learning-vs-deep-learning

Bellan, R.(2022, January 13) Serve Robotics' new autonomous sidewalk delivery robots don't require human assist. TechCrunch. https://techcrunch.com/2022/01/13/serve-robotics-new-autonomous-sidewalk-delivery-robots-dont-require-human-assist/

Mu, G., Xinyu, Z., Deyi, L., Tianlei, Z., Lifeng, A.(2015). Traffic light detection and recognition for autonomous vehicles. The Journal of China Universities of Posts and Telecommunications, 22, 50-56. https://doi.org/10.1016/S1005-8885(15)60624-0

Corovic, A., Ilic, V., Marijan, M. (2018). The Real-Time Detection of Traffic Participants Using YOLO Algorithm. Telfor 2018. https://doi.org/10.1109/TELFOR.2018.8611986

Bach, M., Stumper, D., Dietmayer, K. (2018). Deep Convolutional Traffic Light Recognition for Automated Driving. 2018 21st International Conference on Intelligent Transportation Systems (ITSC). https://doi.org/10.1109/ITSC.2018.8569522

Dodge, S., Karam, L. (2016). Understanding How Image Quality Affects Deep Neural Networks. Arizona State University. https://doi.org/10.1109/QoMEX.2016.7498955

LAMPIRAN

JADWAL KEGIATAN

No	Nama Kegiatan	Minggu ke-															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	Studi Pustaka																
2	Simulasi																
3	Analisa																
4	Pengolahan data																
5	Pelaporan kemajuan																
6	Penyusunan laporan																

BIODATA PENULIS

Penulis merupakan anak pertama dari 2 bersaudara kelahiran Jakarta, 25 Juni 1998. Penulis menempuh pendidikan Tingkat Dasar di SD Mater Dei Pamulang, kemudian melanjutkan Sekolah Menengah Pertama di SMP Mater Dei Pamulang, kemudian melanjutkan Sekolah Menengah Atas di SMAN 2 Tangerang Selatan. Pada tahun 2016 penulis diterima di program S-1 Departemen Teknik Mesin, Fakultas Teknologi Industri dan Rekayasa Sistem, Institut Sepuluh Nopember dan terdaftar sebagai mahasiswa dengan NRP 02111640000104. Penulis aktif dalam kepanitiaan kegiatan di lingkungan kampus ITS, seperti menjadi Panitia dokumentasi Mechanical City 2017, Koordinator wilayah Tangerang Mechanic Skill Competition, dan kegiatan kepanitiaan lainnya. Sehubungan dengan hasil penelitian yang dilakukan, untuk menghubungi penulis dalam rangka pemenuhan kritik dan saran dari pembaca, dapat dilakukan melalui email: bjbbeat@gmail.com.