IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA *REAL TIME* BERBASIS YOLOV8

SKRIPSI

MUH. YUSUF SYAM

H071191044

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2024

LEMBAR PERNYATAAN KEOTENTIKAN

Saya yang bertanda tangan di bawah ini menyatakan dengan sungguh-sungguh bahwa skripsi yang saya buat dengan judul:

IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA REAL TIME BERBASIS YOLOV8

adalah benar hasil karya saya sendiri, bukan hasil plagiat dan belum pernah dipublikasikan dalam bentuk apapun

Makassar, 19 Januari 2024

Muh. Yusuf Syam
NIM. H071191044

IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA *REAL TIME* BERBASIS YOLOV8

Disetujui oleh:

Pembimbing Utama	Pembimbing Pertama
Dr. Eng. Armin Lawi, S.Si., M.Eng.,	**A. Muh. Amil Siddik, S.Si., M.Si.**
NIP. 197204231995121001	**NIP. 199110032019031015**

Ketua Program Studi

Dr. Khaeruddin, M.Sc.

NIP. 196509141991031003

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh

Nama	:	Muh. Yusuf Syam
NIM	:	H071191044
Program Studi	:	Sistem Informasi
Judul Skripsi	:	IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA REAL TIME BERBASIS YOLOV8

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

DEWAN PENGUJI

				Tanda Tangan
1.	Ketua	:	Dr. Eng. Armin Lawi, S.Si., M.Eng.	(.....)
2.	Sekretaris	:	A. Muh. Amil Siddik, S.Si., M.Si.	(.....)
3.	Anggota	:	Dr. Muhammad Hasbi, M.Sc.	(.....)
4.	Anggota	:	Rozalina Amran, S.T., M.Eng.	(.....)

Ditetapkan di : Makassar

Tanggal : 19 Maret 2024

KATA PENGANTAR

Bismillahirrahmanirrahim. Puji syukur penulis panjatkan ke hadirat Allah Subhanahu Wata'ala atas limpahan rahmat, hidayah serta karunia-Nya sehingga penulisan skripsi yang berjudul "**IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA *REAL TIME* BERBASIS YOLOV8**" dapat terselesaikan dengan baik. Shalawat serta salam senantiasa tercurah kepada junjungan kita Nabi Muhammad sallallahu 'alaihi wasallam yang telah menjadi suri tauladan bagi kita semua.

Skripsi ini dibuat dan diajukan untuk memenuhi syarat dalam menyelesaikan Pendidikan strata satu (S1) Sarjana Komputer di Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin. Penulis berharap, skripsi yang ditulis dapat memberikan kontribusi yang bermanfaat bagi perkembangan ilmu pengetahuan dan teknologi, khususnya dalam bidang machine learning. Semoga hasil penelitian ini dapat menjadi pijakan untuk penelitian lebih lanjut yang mendalam dan bermanfaat bagi kemajuan bangsa dan negara.

Segenap keluarga, para dosen pembimbing dan penguji, teman-teman, serta semua pihak yang turut berperan dalam perjalanan penulisan skripsi ini, menjadi bagian yang tak terpisahkan dalam mencapai kesuksesan. Dengan kerendahan hati, penulis mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan dukungan, bimbingan dan motivasi dalam proses penyusunan skripsi ini:

1. Ayahanda tercinta **Sampara ST. MM.,** dan ibunda tersayang **Mitha Fahriah Hakim** yang telah membesarkan, membimbing dan mendidik penulis, sehingga dapat menyelesaikan studi dengan baik. Tanpa dukungan, bantuan dan doa mereka penulisan skripsi ini tidak akan mungkin terselesaikan dengan baik.
2. Rektor Universitas Hasanuddin, Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M. Sc.,** beserta jajarannya.
3. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, **Dr. Eng. Amiruddin, S.Si., M.Si.,** beserta jajarannya.

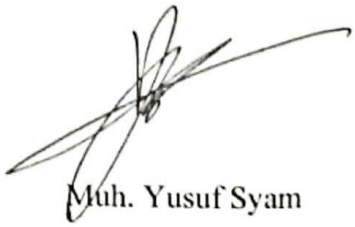
4. Ketua Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin, Bapak **Prof. Dr. Nurdin, S. Si., M. Si.,**.
5. Ketua Program Studi Sistem Informasi, Bapak **Dr. Khaeruddin, M.Sc.,** atas seluruh ilmu dan arahan bagi penulis selama masa studi penulis.
6. Bapak **Dr. Eng. Armin Lawi, S.Si., M.Eng.,** sebagai dosen pembimbing utama yang telah meluangkan waktunya serta memberikan bimbingan dan arahan yang berharga selama proses penyusunan skripsi.
7. Bapak **A. Muh. Amil Siddik, S.Si., M.Si.,** selaku dosen pembimbing pertama yang telah memberikan arahan, bimbingan, dan panduan dengan penuh kesabaran serta dedikasi yang tinggi selama proses penyusunan skripsi.
8. Bapak **Dr. Muhammad Hasbi, M.Sc.,** selaku dosen penguji pertama, yang telah meluangkan waktu mulai dari seminar proposal hingga sidang skripsi untuk memberikan saran dan masukan yang berharga selama penulisan skripsi.
9. Ibu **Rozalina Amran, S.T., M.Eng.,** sebagai dosen penguji kedua sekaligus penasehat akademik, yang telah senantiasa membantu, membimbing serta memberikan arahan selama masa studi penulis khususnya dalam masa penyusunan skripsi ini.
10. Seluruh **Dosen, Staf Pengajar,** serta **Pegawai Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin** atas ilmu, bantuan, kemudahan dan keramahan selama penulis menempuh proses perkuliahan.
11. **Adik-adik penulis**, Ayyub, Aan, Aira, Faraz dan Shanum yang selalu memberikan dukungan, semangat, dan keceriaan selama masa pengerjaan skripsi.
12. Sahabat "**Fokus Hidup**" Andi Ilhamsyah, Alif Setya, Bayu Ajid, Fajri Rasid, Fatwa Anugrah, Fauzi Asham, Muhammad Takdim, Muammar Ahlan, Muhammad Ikhsan, Rafly Masloman, Richard Enrico, Silverius Sony, Taufiq Goe dan Theodarryl yang saling memberikan dukungan,

semangat dan informasi kepada satu sama lain selama masa perkuliahan hingga selesainya penulisan skripsi penulis.

13. Keluarga "**Lambe Turah**" sekaligus sepupu-sepupu penulis, yaitu Halik, Bida, Tina, Lulu, Udin dan Ume, yang senantiasa memberikan dukungan moril dan menghibur penulis selama masa penyusunan skripsi.
14. Teman-teman **KKNT 109 ITTG Bantaeng Posko 2**, yaitu Jailani, Tia, Yusran, Puspita, Albar, Mukhlis, Ilma, Jufri dan Rizqal yang selalu kompak memberikan dorongan kepada satu sama lain untuk menyelesaikan studi.
15. Seluruh teman-teman **Program Studi Sistem Informasi Angkatan 2019** yang telah menjadi rekan seperjuangan dari awal sampai akhir masa perkuliahan di Kampus Merah tercinta.
16. Komunitas ***Stack Overflow, Kaggle, Github, GeeksForGeeks, Medium, IndonesiaAI*** atas kontribusinya sebagai sumber pembelajaran dan tempat tanya jawab yang mendukung proses penyelesaian tugas perkuliahan dan penulisan skripsi ini.

Akhir kata, penulis berharap Tuhan Yang Maha Esa melimpahkan berkat-Nya kepada semua pihak yang telah mendukung dan membantu penulisan skripsi ini. Peneliti menyadari bahwa pembuatan penelitian skripsi ini masih banyak kekurangan dan kelemahaannya, baik dari materi maupun teknik penyajiannya. Oleh karena itu, kritik dan saran yang membangun sangat diharapkan penulis untuk menambah kesempurnaan penelitian skripsi ini.

Makassar, 16 Maret 2024



Muh. Yusuf Syam

PERSYARATAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	:	Muh. Yusuf Syam
NIM	:	H071191044
Program Studi	:	Sistem Informasi
Departemen	:	Matematika
Fakultas	:	Fakultas Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	:	Skripsi

Demi pengembangan ilmu pengetahuan, menyutujui untuk memberikan kepada Universitas Hasanuddin Hak Predikator Royalti Nonekslusif (*Non-exclusive Royalty-Free Right*) atas tugas akhir saya yang berjudul:

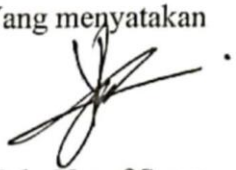
"IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA REAL TIME BERBASIS YOLOV8"

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal diatas, maka pihak Universitas Hasanuddin berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat dan mempublikasikan tugas akhir pemilik Hak Cipta.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di Makassar pada 23 Januari 2024

Yang menyatakan



Muh. Yusuf Syam

ABSTRAK

Kemacetan lalu lintas adalah masalah serius dengan dampak merata pada masyarakat, lingkungan, dan ekonomi. Untuk mengatasinya, banyak kota menerapkan kebijakan ganjil-genap untuk mengatur penggunaan jalan berdasarkan nomor plat kendaraan. Penelitian ini bertujuan untuk mengembangkan sistem deteksi ganjil-genap otomatis pada plat kendaraan dengan akurasi tinggi, menggantikan metode manual yang umumnya digunakan. Sistem menggunakan ESP32-CAM untuk mengambil gambar lalu lintas, algoritma YOLOv8 untuk mendeteksi plat kendaraan, dan OCR untuk mengenali karakter plat. Hasil deteksi disimpan di *website* yang dibangun menggunakan ReactJs, FastAPI, dan MySQL. Uji coba menunjukkan bahwa sistem berhasil mendeteksi 8 plat kendaraan dengan total *levenshtein distance* 7 pada 8 sampel uji, dengan rata-rata *levenshtein distance* 0.875, dan akurasi pembacaan status ganjil-genap sebesar 100%. Dari 64 karakter plat kendaraan, 57 terbaca dengan benar, menghasilkan akurasi pembacaan karakter plat kendaraan sebesar 89%, selanjutnya sistem. Hasil penelitian menunjukkan bahwa pengembangan sistem deteksi ganjil-genap pada plat kendaraan menggunakan ESP32-CAM, algoritma YOLOv8, dan OCR berhasil dengan akurasi tinggi.

Kata Kunci: Kemacetan Lalu Lintas, Kebijakan Ganjil-Genap, *You Only Look Once*, ESP32-CAM, *Optical Character Recognition*, ReactJs, FastAPI, MySQL, *Levenshtein Distance*

ABSTRACT

Traffic congestion is a serious issue with widespread impacts on society, the environment, and the economy. To address this, many cities implement odd-even policies to regulate road usage based on vehicle license plate numbers. This research aims to develop an automated odd-even detection system for vehicle license plates with high accuracy, replacing the commonly used manual methods. The system utilizes ESP32-CAM to capture traffic images, YOLOv8 algorithm for vehicle plate detection, and OCR for plate character recognition. Detected information is stored on a website built using ReactJs, FastAPI, and MySQL. Testing shows that the system successfully detected 8 vehicle plates with a total levenshtein distance of 7 out of 8 test samples, with an average levenshtein distance of 0.875, and odd-even status reading accuracy of 100%. Out of 64 plate characters, 57 were correctly read, resulting in an accuracy of 89%. The research findings indicate that the development of an odd-even detection system for vehicle plates using ESP32-CAM, YOLOv8 algorithm, and OCR was successful with high accuracy.

Keywords: Traffic Congestion, Odd-Even Policy, You Only Look Once, ESP32-CAM, Optical Character Recognition, ReactJs, FastAPI, MySQL, Levenshtein Distance

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Kemacetan lalu lintas adalah masalah serius yang memiliki dampak luas terhadap masyarakat, lingkungan, dan ekonomi. Kemacetan meningkatkan polusi udara, berdampak pada kesehatan manusia, serta menghambat pertumbuhan ekonomi (Levinson & Krizek, 2008). Salah satu faktor yang menyebabkan kemacetan adalah tingginya jumlah kendaraan yang beroperasi di jalan-jalan perkotaan.

Untuk mengatasi masalah kemacetan, banyak kota besar menerapkan kebijakan ganjil-genap, yang mengatur penggunaan jalan berdasarkan nomor plat kendaraan. Namun, penerapan kebijakan ganjil-genap sering kali masih dilakukan secara manual oleh petugas lalu lintas, yang dapat menyebabkan kesalahan, ketidakakuratan, dan biaya operasional yang tinggi.

Dalam konteks ini, teknologi *computer vision* dan *machine learning* menawarkan solusi yang menjanjikan. Penelitian "IMPLEMENTASI SISTEM DETEKSI PLAT NOMOR GANJIL-GENAP SECARA *REAL TIME* BERBASIS YOLOV8" bertujuan untuk mengembangkan sistem otomatis yang dapat mendeteksi ganjil-genap pada plat kendaraan dengan akurasi tinggi.

Dengan menggunakan model objek deteksi *You Only Look Once* versi 8 (YOLOv8), sistem akan mendeteksi plat kendaraan dalam gambar dengan cepat dan akurat. Selanjutnya, menggunakan teknik *Optical Character Recognition* (OCR), sistem akan mengenali karakter nomor plat kendaraan pada gambar. Dengan mendeteksi dan mengenali nomor plat dengan presisi, sistem akan dapat menentukan apakah kendaraan tersebut termasuk dalam kategori ganjil atau genap.

Diharapkan bahwa implementasi sistem otomatis ini akan membantu mengoptimalkan penerapan kebijakan ganjil-genap dengan mengurangi *human error* dan meningkatkan efisiensi penegakan aturan lalu lintas. Dengan demikian, penelitian ini dapat berkontribusi dalam mengatasi masalah kemacetan dan meningkatkan kualitas lalu lintas perkotaan.

Sebelumnya telah ada beberapa penelitian serupa yang menyinggung tentang deteksi dan rekognisi plat kendaraan. Penelitian berjudul "*An efficient automated vehicle license plate recognition system under image processing*" yang dilakukan oleh (Islam et al., 2022), mengusulkan suatu sistem rekognisi plat kendaraan yang terstruktur dalam empat tahap utama, melibatkan praproses gambar *input*, ekstraksi wilayah plat kendaraan, segmentasi karakter plat, dan rekognisi karakter plat pada tahap akhir. Pada tahap praproses, format warna gambar *input* diubah dari RGB menjadi format *grayscale*, dilanjutkan dengan penerapan *bilateral filter* untuk mengurangi *noise*. Ekstraksi wilayah plat kendaraan dilakukan melalui operasi morfologi dan *filter* Sobel, sementara segmentasi karakter memanfaatkan *connected component analysis* dan pendekatan *bounding box*. Rekognisi karakter plat dilakukan dengan *template matching*, yaitu teknik yang membandingkan setiap karakter dengan *database template* yang disimpan sebelumnya. Dengan implementasi langkah-langkah tersebut, sistem ini berhasil mengenali karakter-karakter pada plat kendaraan, dan memberikan *output* hasil prediksi melalui notepad.

Penelitian terkait berikutnya dalam pengembangan sistem deteksi plat ganjil-genap berjudul "YOLO v5 untuk Deteksi Nomor Kendaraan di DKI Jakarta" yang dilakukan oleh (Illmawati & Hustinawati, 2023). Penelitian ini bertujuan untuk mendeteksi plat kendaraan di DKI Jakarta, lalu mengkategorikannya menjadi genap atau ganjil berdasarkan karakter dari plat kendaraan. Hasil dari penelitian ini berupa pengujian terhadap 25 video, yang menghasilkan rata-rata akurasi pendeteksian objek sebesar 92.38%, rata-rata akurasi rekognisi karakter dalam plat nomor memiliki sebesar 95.45%, dan rata-rata akurasi pada tingkat keberhasilan program untuk menentukan kategori dari plat kendaraan sebesar 97.2%.

Selanjutnya penelitian yang berjudul "*Object Tracking* Menggunakan Algoritma *You Only Look Once* (YOLO) v8 untuk Menghitung Kendaraan (2023)" yang dilakukan oleh (Nurhaliza et al., 2023). Penelitian ini bertujuan untuk mengatasi kendala dalam pengumpulan data lalu lintas, khususnya terkait pencacahan jenis dan jumlah kendaraan. Penelitian ini menggunakan pendekatan terstruktur dengan tahapan *AI Project Cycle*, melibatkan *problem scoping*, *data*

acquisition, data exploration, modeling, dan *evaluation* menggunakan *confusion matrix*. Algoritma YOLOv8 digunakan untuk mendeteksi jenis dan menghitung jumlah kendaraan. DeepSORT sebagai algoritma pelacakan objek. Pengujian model dilakukan pada data video dengan membandingkan jumlah kendaraan yang diprediksi oleh sistem dengan jumlah kendaraan aktualnya. Akurasi yang diperoleh yaitu sebesar 86%, *precision* sebesar 86%, *recall* sebesar 86%, dan nilai *F1-Score* sebesar 85%.

Penelitian yang berjudul "*Improved Face Detection Accuracy Using HAAR Cascade Classifier Method and ESP32-CAM For IOT-Based Home Door Security*" yang dilakukan oleh (Muhammad et al, 2022) bertujuan untuk meningkatkan keamanan pintu rumah menggunakan sistem *face recognition* berbasis ESP32-CAM. Metodologi penelitian ini melibatkan pengumpulan data dari 12 orang dengan 6 di antaranya terdaftar dalam *database*. Pengujian dilakukan pada jarak 30 cm, 40 cm, dan 50 cm dengan intensitas cahaya sebesar 130 lux. ESP32-CAM digunakan sebagai perangkat *face recognition*. Hasil pengujian *face recognition* menggunakan metode *Haar Cascade Classifier*, menunjukkan kemampuan untuk membedakan wajah yang terdaftar dan tidak terdaftar. Akurasi rata-rata *face recognition* pada jarak 30 cm, 40 cm, dan 50 cm dengan intensitas cahaya 130 lux adalah 96,6%. Waktu rata-rata yang dibutuhkan dari proses pengambilan sampel wajah hingga *face recognition* oleh sistem adalah 21,50 detik. Sistem keamanan yang dikembangkan menggunakan ESP32-CAM memiliki beberapa fitur, termasuk kemampuan membuka kunci otomatis jika wajah yang terdaftar terdeteksi, membuka kunci menggunakan sidik jari, mengendalikan kunci pintu melalui *smartphone*, dan mengirimkan pemberitahuan kepada pemilik rumah jika wajah yang tidak dikenal terdeteksi.

Penelitian yang berjudul "*Real-Time Traffic Sign Detection and Classification Using Machine Learning and Optical Character Recognition*" yang dilakukan oleh (Ciuntu & Ferdowsi, 2020). Pada penelitian ini, para peneliti membangun sebuah sistem kompleks untuk deteksi dan klasifikasi rambu lalu lintas secara *real-time* menggunakan kombinasi *Convolutional Neural Network* (CNN) dan *Custom Optical Character Recognition* (OCR). Melalui penggunaan CNN, penelitian ini berhasil mencapai tingkat akurasi deteksi dan klasifikasi yang

baik. Sistem yang diusulkan berhasil mendeteksi 92,8% rambu lalu lintas dengan benar dengan rata-rata waktu pemrosesan sebesar 251.527 detik, meskipun menghadapi beberapa kesalahan deteksi sebesar 10,1%.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah diuraikan diatas, maka dapat diambil rumusan masalah sebagai berikut:

1. Bagaimana merancang sistem deteksi plat nomor ganjil genap secara *real-time* yang komprehensif?
2. Bagaimana merancang dan melatih model *object detection* YOLOv8 dan *Optical Character Recognition* (OCR) untuk mendeteksi plat kendaraan?
3. Bagaimana merancang dan mengembangkan sebuah *website* dengan menggunakan *backend* FastAPI dan *frontend* ReactJS?
4. Bagaimana mengevaluasi kinerja sistem deteksi plat?

1.3 Batasan Masalah

Berdasarkan masalah diatas, maka batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Sistem akan difokuskan pada deteksi dan pengenalan nomor plat dari mobil pribadi.
2. Sistem akan difokuskan pada deteksi dan pengenalan nomor plat kendaraan yang berlaku di Negara Indonesia.
3. Sistem akan dirancang untuk bekerja dalam batas waktu tertentu dengan kondisi pencahayaan yang cukup normal dan tidak ekstrim.
4. Kamera yang digunakan pada sistem yang dirancang adalah ESP32-CAM.
5. ESP32-CAM akan diposisikan di sepanjang jalan raya menghadap ke arah datangnya kendaraan, ditempatkan pada elevasi dan sudut tertentu.
6. Algoritma deteksi objek plat kendaraan yang digunakan adalah YOLOv8.
7. *Website* dibangun menggunakan ReactJs dan FastAPI.

1.4 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian adalah sebagai berikut:

1. Untuk merancang sistem deteksi plat nomor ganjil genap secara *real-time* yang komprehensif.
2. Untuk merancang dan melatih model *object detection* YOLOv8 dan *Optical Character Recognition* (OCR) untuk mendeteksi plat kendaraan.

3. Untuk merancang dan mengembangkan sebuah *website* dengan menggunakan *backend* FastAPI dan *frontend* ReactJS.
4. Untuk mengevaluasi kinerja sistem deteksi plat kendaraan.

BAB II
TINJAUAN PUSTAKA

2.1 Kebijakan Plat Ganjil Genap

Kebijakan ganjil-genap adalah salah satu langkah yang diambil oleh pemerintah kota untuk mengurangi kemacetan lalu lintas dan polusi udara. Dalam kebijakan ini, kendaraan dengan nomor plat ganjil hanya diperbolehkan beroperasi pada hari ganjil, sedangkan kendaraan dengan nomor plat genap hanya diperbolehkan pada hari genap (Putra, 2019). Hari yang dianggap sebagai hari ganjil adalah hari dengan tanggal berakhiran nomor ganjil (1, 3, 5, 7, 9), sementara hari genap adalah hari dengan tanggal berakhiran nomor genap (2, 4, 6, 8, 0). Di Indonesia, kebijakan ini utamanya diterapkan di ibu kota Jakarta, dan beberapa kota besar lainnya seperti kota Bogor, Bandung dan Cirebon. Kebijakan ini juga telah diterapkan di beberapa kota besar di seluruh dunia, termasuk kota Meksiko, Paris, Bogota dan Beijing.

Beberapa penelitian telah dilakukan untuk mengevaluasi manfaat dan efektivitas kebijakan ganjil-genap. Studi oleh (Putri et al., 2021) yang berjudul "Simulasi Dampak Rencana Penerapan Skema Ganjil Genap di Kota Bekasi" menunjukkan dengan adanya simulasi penerapan kebijakan ganjil genap di Kota Bekasi, dapat mengurangi volume lalu lintas, meningkatkan rata-rata kecepatan kendaraan, meningkatkan minat penggunaan transportasi umum dan menurunkan waktu tempuh perjalanan rata-rata pada ruas jalan uji coba.

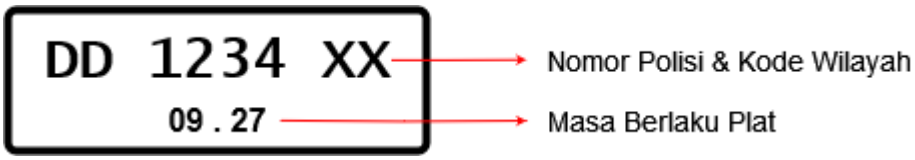
Meskipun kebijakan ganjil-genap telah menunjukkan manfaatnya dalam mengurangi kemacetan dan polusi udara, kebijakan ini juga dihadapkan pada beberapa tantangan dan kontroversi. Beberapa kritikus berpendapat bahwa kebijakan ini dapat meningkatkan jumlah kendaraan pribadi, karena pemilik kendaraan akan cenderung membeli kendaraan tambahan untuk mengatasi larangan beroperasi pada hari tertentu. Selain itu, ada juga masalah penegakan kebijakan dan kesadaran masyarakat terkait implementasi kebijakan ini.

2.1.1 Plat Kendaraan Negara Indonesia

Plat kendaraan adalah tanda identifikasi atau registrasi kendaraan yang terdiri atas kombinasi huruf dan angka yang sifatnya unik dan khusus untuk kendaraan tersebut (Barnabas, 2022). Plat kendaraan berfungsi sebagai tanda

identifikasi pada kendaraan mobil ataupun motor serta kendaraan umum lainnya. Plat kendaraan juga menunjukkan informasi mengenai kendaraan seperti model, tahun pembuatan, wilayah kendaraan dan lainnya (Baladewa, 2023). Plat kendaraan wajib dipasang di bagian depan dan belakang kendaraan dan harus sesuai aturan serta tidak boleh ditutupi atau terhalang benda lain. Jenis plat nomor kendaraan ada banyak jenisnya, meski yang mungkin umum diketahui oleh banyak orang adalah warna yang sering dijumpai di jalan seperti hitam dan kuning serta merah. Jenis plat nomor kendaraan yang ada antara lain: plat nomor kendaraan dengan dasar hitam dan tulisan putih, plat nomor kendaraan dengan dasar kuning dan tulisan hitam, plat nomor kendaraan dengan dasar merah dengan tulisan putih, dan plat nomor kendaraan dengan dasar putih dengan tulisan biru (Yuniharto, 2020).

Format plat nomor kendaraan di setiap negara berbeda-beda, dan khususnya Indonesia terdiri dari dua baris. Baris pertama berisi kode wilayah, yang terdiri dari satu atau dua huruf, diikuti oleh nomor polisi berupa angka, dan diakhiri dengan seri akhir yang menandakan wilayah terkait yang terdiri dari satu sampai tiga huruf. Sedangkan pada baris kedua, terdapat informasi masa berlaku plat nomor, yang tersusun dalam format bulan dan tahun (Barnabas, 2022). Sebagaimana pada Gambar 2.1.



Gambar 2.1 Format Plat Kendaraan Negara Indonesia

Daftar Provinsi dan kode wilayah plat kendaraannya ditunjukkan pada tabel 2.1.

Tabel 2.1 Daftar Provinsi Negara Indonesia dan Kode Wilayahnya

Provinsi	**Seri Akhir Plat Nomor**
Aceh	BL
Sumatera Utara	BB
Sumatera Barat	BA

Riau	BM
Jambi	BH
Bengkulu	BD
Sumatera Selatan	BG
Kepulauan Bangka	BN
Lampung	BE
Banten	A
Jakarta	B
Jawa Barat	D
Jawa Tengah	H
Yogyakarta	AB
Jawa Timur	N
Bali	DK
Nusa Tenggara Barat	EA
Nusa Tenggara Timur	EB
Kalimantan Barat	KB
Kalimantan Tengah	KH
Kalimantan Selatan	DA
Kalimantan Timur	KT
Sulawesi Utara	DB
Sulawesi Tengah	DN
Sulawesi Selatan	DD
Sulawesi Tenggara	DT
Gorontalo	DM
Maluku	DE
Maluku Utara	DG
Papua Barat	PB

Papua	PA

Saat ini, format plat nomor kendaraan pribadi permanen di Indonesia adalah warna dasar putih dengan tulisan berwarna hitam. Jenis plat kendaraan yang ada antara lain: plat nomor kendaraan dengan dasar hitam dan tulisan putih, plat nomor kendaraan dengan dasar kuning dan tulisan hitam, plat nomor kendaraan dengan dasar merah dengan tulisan putih, dan plat nomor kendaraan dengan dasar putih dengan tulisan biru (Yuniharto, 2020).

Gambar 2.2 Jenis Plat Kendaraan di Negara Indonesia

2.2 *Deep Learning*

Kecerdasan buatan atau *artificial intelligence* (AI) mengacu pada pengembangan teknologi yang mampu meniru kemampuan berfikir dan bertindak manusia (Maryani et al., 2023). AI membuat agar mesin (komputer) dapat melakukan pekerjaan seperti dan sebaik yang dilakukan oleh manusia, seperti pengenalan suara, *Natural Language Processing* (pemrosesan bahasa alami), perencanaan, serta pengambilan keputusan. Dalam upaya ini, *machine learning* (ML) muncul sebagai cabang penting dari kecerdasan buatan.

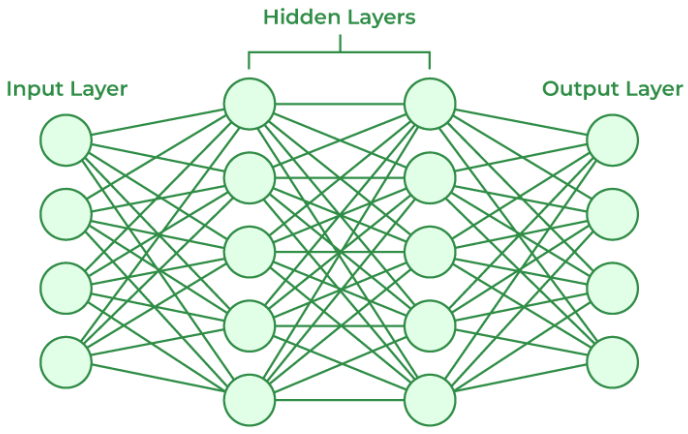
Selama beberapa dekade terakhir, bidang ML telah melahirkan berbagai kemajuan luar biasa dalam pembelajaran canggih algoritma dan teknik pra-pemrosesan yang efisien. Satu dari kemajuan ini adalah arsitektur *Artificial Neural Network* (ANN) yang semakin dalam dengan kemampuan pembelajaran yang meningkat, yang dinamakan sebagai *deep learning*.

Deep learning adalah cabang dari *machine learning* yang mengembangkan algoritma dan model komputasi yang terinspirasi dari struktur dan fungsi otak

manusia yang kompleks. *Deep learning* fokus pada pembelajaran representasi hierarkis dari data dengan menggunakan jaringan syaraf tiruan yang dalam (*deep neural networks*) untuk menghasilkan prediksi atau klasifikasi yang akurat. Teknik ini telah berhasil diterapkan dalam berbagai bidang seperti pengenalan gambar, pemrosesan bahasa alami, dan analisis data (Lecun et al., 2015). Mayoritas dari model *deep learning* menggunakan struktur ANN sebagai kerangka kerjanya. ANN bisa memiliki sekitar 2 - 200 atau lebih lapisan tersembunyi, istilah "*deep*" pada umumnya merujuk pada jumlah lapisan tersembunyi dalam jaringan saraf tersebut.

2.2.1 *Artificial Neural Network*

Artificial Neural Network (ANN) atau jaringan saraf tiruan adalah sebuah teknik atau pendekatan pengolahan informasi yang terdiri dari sejumlah besar elemen pemrosesan informasi (*neuron*) yang saling terhubung dan bekerja bersama-sama untuk menyelesaikan sebuah masalah tertentu, yang pada umumnya adalah masalah klasifikasi atau prediksi. Model *deep learning* ini mengambil inspirasi dari kedalaman struktur otak manusia, yang belajar dengan memproses konsep mulai dari yang paling dasar hingga yang paling kompleks (Sarker, 2021). Konsep ini dapat diilustrasikan dalam Gambar 2.3, yang menunjukkan jaringan saraf yang terdiri dari berbagai lapisan seperti *input layer*, *hidden layer* dan *output layer*.



Gambar 2.3 Artificial *Neural Network*

Berikut penjelasan dari *input layer*, *hidden layers*, dan *output layer*:

1. *Input Layer*

- *Input layer* merupakan layer pertama dari jaringan saraf tiruan.
- Fungsi dari *input layer* adalah menerima input dari *dataset* atau lingkungan eksternal dan meneruskannya ke *hidden layer* untuk diproses lebih lanjut.
- Setiap *neuron* pada *input layer* mewakili fitur atau variabel *input* dari *dataset*.
- *Neuron* pada *input layer* tidak melakukan komputasi atau transformasi data, melainkan hanya meneruskan *input* ke *hidden layer*.

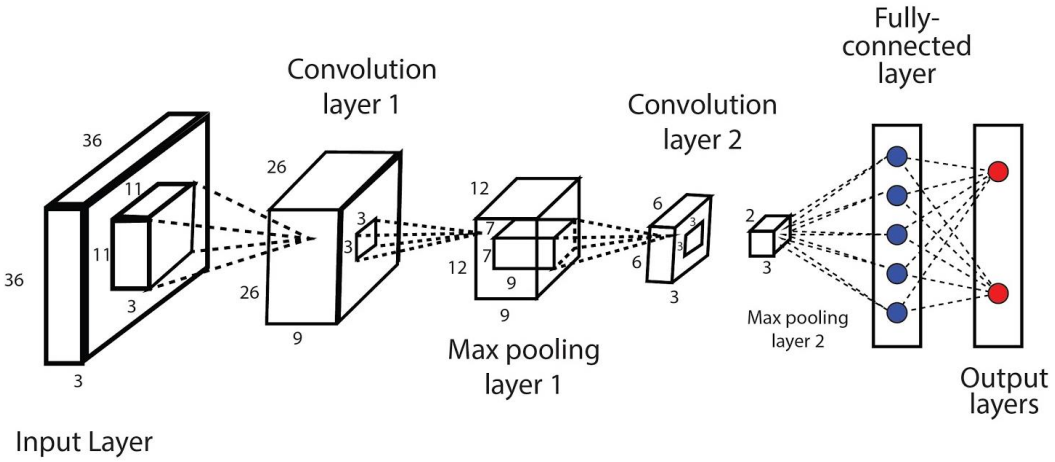
2. *Hidden Layer*
   - *Hidden layer* merupakan *layer* di antara *input layer* dan *output layer.*
   - Fungsi dari *hidden layer* adalah melakukan komputasi dan transformasi terhadap *input* yang diterima dari *input layer.*
   - *Hidden layer* memiliki kemampuan untuk mengekstrak fitur-fitur yang kompleks dari input dan melakukan representasi data yang lebih abstrak.
   - Jumlah *hidden layer* dan jumlah *neuron* di setiap *hidden layer* dapat bervariasi tergantung pada kompleksitas masalah yang ingin diselesaikan.
3. *Output Layer*
   - *Output layer* merupakan *layer* terakhir dari jaringan saraf tiruan.
   - Fungsi dari *output layer* adalah menghasilkan *output* berdasarkan hasil komputasi yang dilakukan oleh *hidden layer*.
   - Jumlah *neuron* pada *output layer* tergantung pada jenis masalah yang ingin diselesaikan, misalnya satu *neuron* untuk masalah regresi dan beberapa *neuron* untuk masalah klasifikasi.

2.2.2 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah sebuah jenis *neural network* yang didesain untuk mengolah data dua dimensi, khususnya data citra. CNN termasuk dalam jenis *Deep Neural Network* karena kedalaman jaringannya yang tinggi dan banyak diaplikasikan pada data citra (Suartika et al., 2016).

CNN memiliki kemampuan untuk mengenali pola-pola kompleks dalam data citra dan telah digunakan dalam berbagai aplikasi seperti klasifikasi citra,

identifikasi karakter, dan pengenalan pola (Umam & Hendoko, 2020). CNN juga memiliki keunggulan dalam mempertahankan informasi spasial dari data citra, yang membuatnya lebih sesuai untuk pengolahan data citra dibandingkan dengan jenis *neural network* lainnya seperti *Multi Layer Perceptron* (MLP).



Gambar 2.4 *Convolutional Neural Network*

Secara garis besar, CNN bekerja dengan cara melakukan operasi konvolusi pada data citra dengan *filter* atau kernel tertentu. *Filter* ini akan mengambil sebagian data citra dan mengalikan setiap elemen dengan bobot tertentu. Hasil dari operasi konvolusi ini akan menghasilkan *feature map* yang merepresentasikan fitur-fitur penting dari data citra. *Feature map* ini kemudian akan diolah lebih lanjut melalui *layer-layer* CNN lainnya seperti *pooling layer* dan *fully connected layer* untuk menghasilkan *output* akhir berupa kelas atau label dari data citra.

Pada diagram arsitektur CNN yang terlihat dalam Gambar 2.4, proses klasifikasi dapat dibagi menjadi dua tahap utama, yaitu *feature learning* dan klasifikasi. Dalam konteks ini, *feature learning* mengacu pada proses pembelajaran fitur, sedangkan klasifikasi merujuk pada langkah identifikasi dan penentuan kategori atau kelas.

2.3 *You Only Look Once* (YOLO)

You Only Look Once (YOLO) adalah sebuah algoritma objek deteksi *real-time* yang sangat populer dalam bidang *computer vision*. Algoritma ini dikembangkan oleh Joseph Redmon, Santosh Divvala, Ross Girshick, dan Ali Farhadi pada tahun 2016. Keunggulan utama dari YOLO adalah kemampuannya

untuk mendeteksi objek dalam sebuah gambar dengan cepat dan akurat, bahkan secara *real-time* (Redmon et al., 2016).

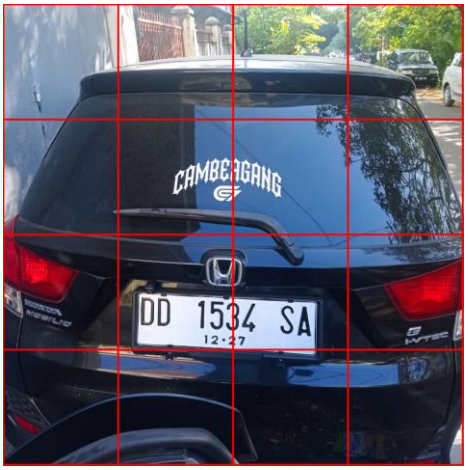
Berbeda dengan metode tradisional deteksi objek yang memerlukan banyak tahap pengolahan seperti RCNN atau RFCN, YOLO menggunakan pendekatan *end-to-end*, yang berarti mendeteksi objek secara langsung dalam satu proses. Dalam YOLO, gambar dibagi menjadi *grid* $S \; X \; S$, dan setiap sel grid bertanggung jawab untuk mendeteksi objek di dalamnya.

Setiap sel *grid* dalam YOLO memprediksi *bounding box* untuk objek yang ada di dalamnya, beserta *confidence score* (skor kepercayaan) untuk mengindikasikan seberapa yakinnya algoritma bahwa *bounding box* tersebut berisi objek. Jika skor kepercayaan lebih tinggi dari batas tertentu, maka objek tersebut dianggap terdeteksi (Redmon et al., 2016). Algoritma YOLO bekerja menggunakan 4 teknik berikut:

A. *Residual Blocks*

Teknik *Residual Blocks* merujuk pada penggunaan blok residual dalam arsitektur YOLO. Blok residual adalah konsep yang diperkenalkan oleh (He et al., 2016) dalam penelitian "*Deep Residual Learning for Image Recognition*". Blok residual memungkinkan model untuk mempelajari perbedaan atau residu antara fitur *input* dan *output*, sehingga memudahkan pembelajaran model yang dalam. Dalam YOLO, penggunaan blok residual membantu dalam pelatihan jaringan yang lebih dalam dan efisien dalam penanganan fitur-fitur yang rumit (He et al., 2016).

Pertama, gambar dibagi menjadi beberapa *grid*. Setiap *grid* memiliki dimensi $S \; X \; S$. Gambar berikut menunjukkan bagaimana gambar dibagi menjadi *grid-grid.*



Gambar 2.5 *Residual Block*

Pada gambar di atas, terdapat banyak sel *grid* dengan dimensi yang sama. Setiap sel *grid* akan mendeteksi objek yang muncul di dalamnya. Dalam kasus ini S atau jumlah *grid* pada panjang dan lebar gambar adalah sebesar 4 *grid*. Selanjutnya setiap sel grid melakukan *localization* dan memprediksi class objek yang dicakupnya bersama dengan nilai *confidence* nya.

B. *Bounding Box Regression*

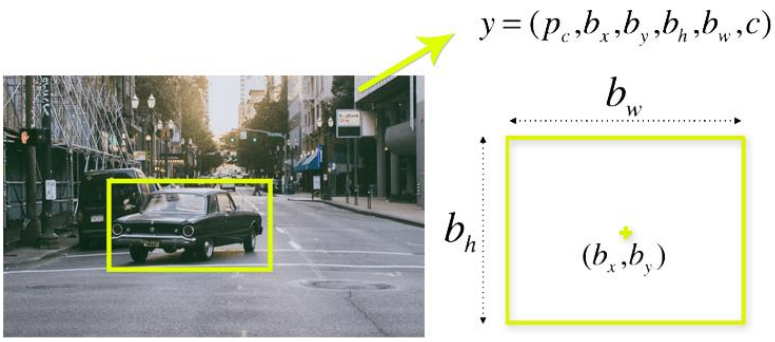
Bounding Box Regression adalah teknik untuk memprediksi koordinat *bounding box* yang melingkupi objek yang ingin dideteksi. Dengan melakukan *Bounding box regression*, model dapat menghasilkan *bounding box* yang lebih tepat sesuai dengan objek yang sebenarnya (Redmon et al., 2016). Setiap *bounding box* Y dalam gambar merepresentasikan suatu vektor yang memiliki setidaknya 6 atribut.

$$Y = [pc, bx, by, bh, bw, c]$$

Berikut penjelasan dari masing-masing atribut dari Y:

- Pc (*probability score*) atau probabilitas suatu sel *grid* memiliki objek di dalamnya.
- Bx, merupakan koordinat x dari titik tengah *bounding box*.
- By, merupakan koordinat y dari titik tengah *bounding box*.
- Bw, merupakan lebar dari *bounding box*.
- Bh, merupakan tinggi dari *bounding box*.
- C, merupakan kelas dari objek yang ingin dideteksi (misalkan mobil, wajah manusia atau hewan tertentu).

Gambar 2.6 menunjukkan suatu *bounding box* pada sebuah mobil yang diwakili dengan garis warna kuning.

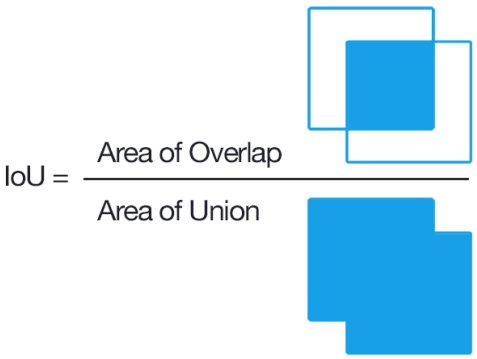


Gambar 2.6 *Bounding Box Regression*

YOLO menggunakan *bounding box regression* untuk memprediksi vektor Y yang berisi atribut skor probabilitas, tinggi, lebar, koordinat tengah, dan kelas objek dari *bounding box.*

C. *Intersection Over Union (IoU)*

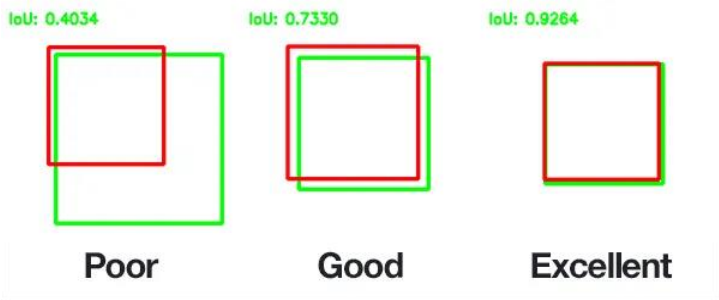
Intersection over Union (IoU) adalah metrik yang digunakan dalam YOLO untuk mengukur sejauh mana *bounding box* yang diprediksi model cocok dengan *bounding box* sebenarnya. IoU dihitung sebagai rasio dari area yang tumpang tindih antara dua *bounding box* terhadap area *intersection* atau area yang disatukan oleh kedua *bounding box* tersebut. Nilai IoU memberikan gambaran seberapa baik *bounding box* yang dihasilkan oleh model dan *bounding box* yang sebenarnya berdampingan. Nilai IoU yang lebih tinggi menunjukkan deteksi yang lebih akurat (Everingham et al., 2009).



Gambar 2.7 Rumus *Intersection Over Union*

IOU akan bernilai 0 jika *bounding box yang* diprediksi tidak beririsan sama sekali dengan *bounding box* sebenarnya, dan akan bernilai 1 kedua *bounding*

box mempunyai koordinat yang sama atau salah satu *bounding box* berada di dalam *bounding box* yang lain.

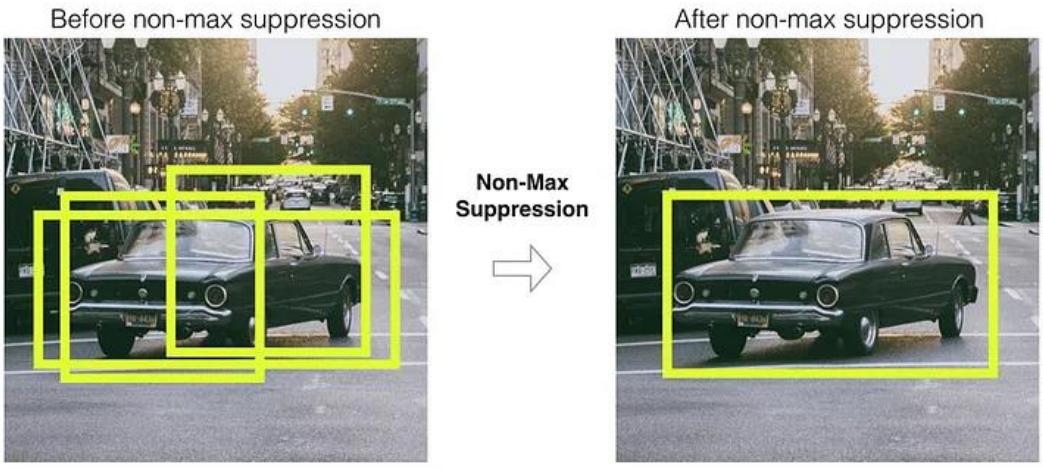


Gambar 2.8 Contoh Nilai IoU dan Kualitasnya

Tujuan utama dari IoU adalah untuk menghilangkan *bounding box* prediksi yang tidak sama dengan *bounding box* sebenarnya, atau *bounding box* yang nilai IoU nya di bawah *threshold* tertentu (biasanya 0.5).

D. *Non-Max Suppression* (NMS)

Non-Max Suppression (NMS) adalah teknik yang digunakan untuk mengatasi masalah deteksi yang berlebihan. Setelah mendeteksi objek, beberapa *bounding box* yang tumpang tindih dapat muncul. NMS digunakan untuk menghapus deteksi berlebihan dengan memilih *bounding box* dengan *confidence score* tertinggi dan menghilangkan *bounding box* lain yang tumpang tindih dengan kotak yang terpilih (Felzenswalb et al., 2010).



Gambar 2.9 *Non-Max Suppression*

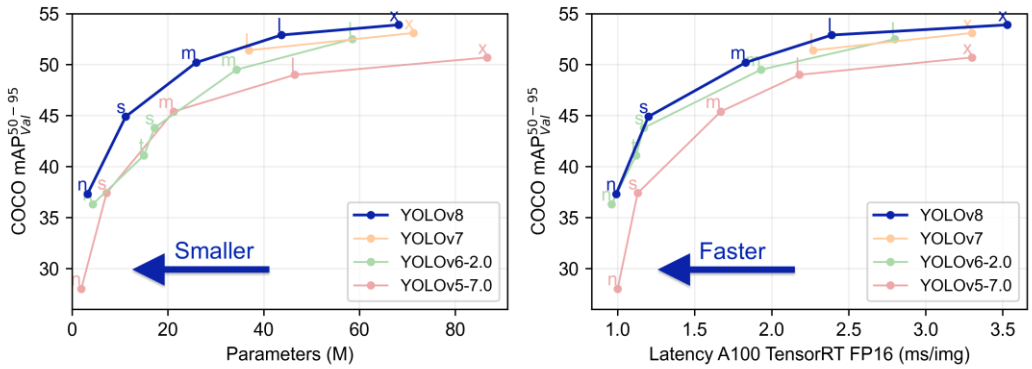
2.3.1 YOLOv8

YOLOv8 adalah iterasi terbaru dari model YOLO dan merupakan *model State of The Art* dalam bidang objek deteksi. YOLOv8 memiliki dukungan

terintegrasi untuk *object detection, classification*, dan *segmentation* (Glenn et al., 2023). YOLOv8 dirilis oleh Ultralytics pada 10 Januari 2023 dan sama seperti pendahulunya, *source code*-nya bersifat *open source*.

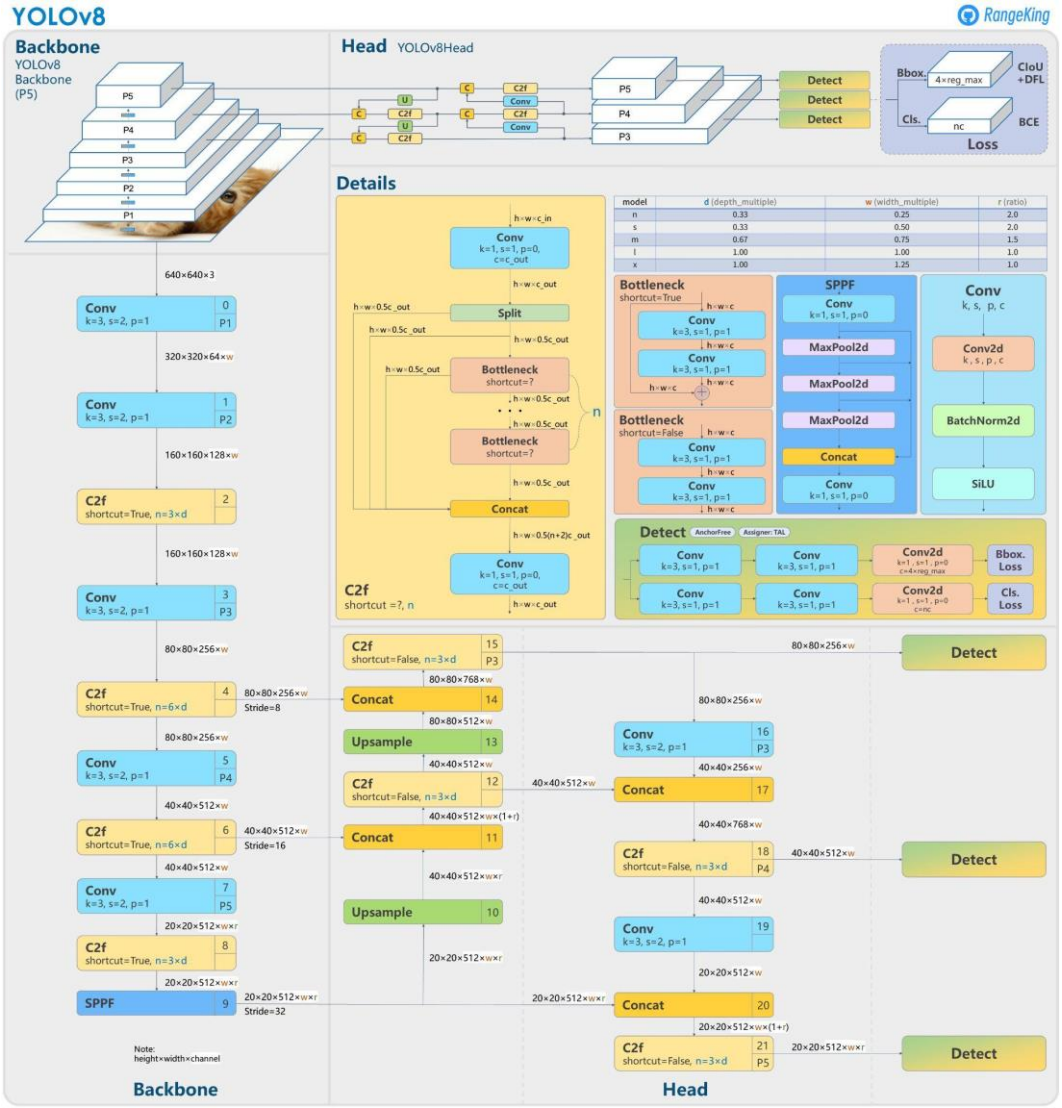
- Perbedaan utama YOLOv8 dari versi-versi sebelumnya meliputi:
- Sistem *Anchor-Free Detection* (deteksi bebas *anchor*) yang lebih fleksibel.
- Perubahan pada blok konvolusi dalam model.
- Penerapan augmentasi data mosaik selama pelatihan

Perubahan utama YOLOv8 termasuk *Anchor-Free Detection*, menggantikan *anchor box* tradisional dengan pendekatan yang lebih dinamis. Augmentasi data mosaik juga diterapkan dalam pelatihan model untuk memperkaya variasi data. YOLOv8 mengungguli versi-versi sebelumnya dalam hal performa, sebagaimana terbukti melalui pengujian terhadap dataset COCO. Hasil pengujian menunjukkan bahwa YOLOv8 mencapai *mean average precision* (mAP) yang lebih tinggi, sambil mempertahankan waktu inferensi yang lebih efisien dan mengurangi jumlah parameter dibandingkan dengan iterasi sebelumnya dari YOLO. Perbandingan performa YOLOv8 dengan versi sebelumnya dapat dilihat pada Gambar 2.10.



Gambar 2.10 Perbandingan Performa YOLOv8

YOLOv8 hadir dengan perubahan untuk pengalaman pengembangan, di mana model ini dikemas sebagai *library* Python, sehingga para *developer* bisa dengan mudah melakukan instalasi dan menggunakan YOLOv8 langsung dari *Command Line Interface* (CLI). Arsitektur YOLOv8 memiliki dua kerangka utama yang tersusun dari *head* dan *backbone*. Berikut ilustrasi tentang arsitektur dari algoritma YOLOv8 pada Gambar 2.11.



Gambar 2.11 Arsitektur YOLOv8

2.4 *Optical Character Recognition* (OCR)

Optical Character Recognition (OCR) adalah algoritma yang digunakan untuk mengenali teks yang tercetak atau ditulis tangan dalam gambar atau dokumen digital, memungkinkan konversi efisien dari teks visual menjadi teks yang dapat diakses oleh komputer (Smith, 2007).

OCR memainkan peran kunci dalam mengotomatisasi proses konversi data teks dari format fisik atau grafis ke dalam bentuk yang dapat diubah dan dicari. Penggunaan OCR telah diterapkan dalam berbagai bidang, termasuk pengolahan dokumen, pengenalan karakter dalam citra medis, dan aplikasi pengenalan tulisan tangan.

2.4.1 EasyOCR

EasyOCR adalah *library open-source* yang mengimplementasikan OCR untuk mendeteksi dan mengenali teks dalam gambar. EasyOCR dirancang untuk menjadi mudah digunakan dan mendukung beberapa bahasa, termasuk bahasa-bahasa yang non-Latin.

EasyOCR dapat digunakan untuk berbagai tujuan, termasuk pengenalan teks dalam citra, pemindaian dokumen, atau integrasi dalam aplikasi yang memerlukan kemampuan OCR. Salah satu keunggulan EasyOCR adalah kemampuannya untuk mengenali berbagai jenis tulisan tangan, serta mendukung berbagai bahasa dan skrip seperti bahasa Inggris, Arab, Korea, Jepang dan Hindi.

EasyOCR dibuat oleh Yongchao Yu, Zhanzhan Cheng, Sheng Jin, dan Xiang Zhou. Pustaka ini pertama kali diunggah sebagai proyek *open-source* di GitHub pada bulan September 2019 dan dirilis *sebagai library open-source* untuk publik pada tanggal 19 Agustus 2020.

2.5 *Internet of Things* (IOT)

Internet of Things (IoT) adalah sebuah teknologi yang memungkinkan adanya pengendalian, komunikasi, dan kerjasama antara berbagai perangkat keras melalui jaringan internet (Junaidi, 2015). IoT memungkinkan objek-objek untuk saling terhubung dan bertukar data melalui jaringan internet, sehingga memungkinkan manusia mudah berinteraksi dengan semua peralatan yang terhubung dengan jaringan internet. IoT telah diterapkan dalam berbagai bidang keilmuan dan industri, seperti dalam bidang ilmu kesehatan, informatika, geografis, dan bidang ilmu lainnya. Contoh penerapan IoT meliputi penggunaan dalam bidang medis untuk konsultasi pasien secara *remote*, penggunaan dalam sistem informasi, dan penggunaan dalam sistem peringatan dini terhadap bencana alam.

2.5.1 ESP32-CAM

Gambar 2.12 ESP32-CAM

ESP32-CAM adalah komponen/modul IoT yang dikembangkan oleh Espressif Systems, dan dikembangkan khusus untuk aplikasi pengambilan gambar dan video serta komunikasi nirkabel. ESP32-CAM memiliki beberapa fitur kunci seperti kamera dan konektivitas *WiFi* atau *Bluetooth*. ESP32-CAM dapat digunakan secara luas di berbagai aplikasi IoT. sangat cocok untuk *home smart devices, industrial wireless control, wireless monitoring, QR wireless identification, wireless positioning system signals* dan aplikasi IoT lainnya (Ipanhar et al., 2022).

A. Fitur

Berikut adalah beberapa fitur utama dari ESP32-CAM:

a. *Microcontroller* ESP32: Modul ini dilengkapi dengan mikrokontroler ESP32, yang memiliki kemampuan komputasi yang tinggi dan dukungan untuk berbagai protokol komunikasi seperti *WiFi* dan *Bluetooth* yang memungkinkan untuk mengontrol kamera dan mengirimkan data gambar atau video melalui jaringan nirkabel.
b. Kamera OV2640: ESP32-CAM dilengkapi dengan kamera OV2640 yang dapat mengambil gambar beresolusi hingga 2 megapiksel (1600x1200 piksel).
c. Slot Kartu *MicroSD*: Modul ini memiliki slot kartu *MicroSD* yang memungkinkan untuk menyimpan gambar dan video yang diambil oleh kamera.

d. Antarmuka GPIO: ESP32-CAM memiliki sejumlah pin GPIO yang dapat digunakan untuk menghubungkan berbagai perangkat tambahan seperti sensor atau aktuator.
e. Konektivitas *WiFi* dan *Bluetooth*: ESP32-CAM memiliki kemampuan untuk terhubung ke jaringan *WiFi* dan *Bluetooth*.
f. Mendukung Arduino: ESP32-CAM dapat diatur dan dikodekan menggunakan Arduino IDE, yang membuatnya mudah digunakan oleh pengembang dengan berbagai tingkat pengalaman.

B. Spesifikasi

Berikut spesifikasi dari Mikrokontroler ESP32-CAM dan Kamera OV2640:

1. Mikrokontroler ESP32-CAM

Tabel 2.2 Spesifikasi Mikrokontroler ESP32-CAM

No.	**Spesifikasi**	**Nilai**
1	Model Modul	ESP32-CAM
2	Ukuran	27 x 40.5 x 4.5 mm
3	*WI-FI*	802.11 b/g/n
4	*Bluetooth*	● BR/EDR ● BLE
5	*RAM*	*520 KB SRAM + 4M PSRAM*
6	*Support TF Card*	Dukungan Maksimal 4G
7	Jumlah IO *Port*	9 *Port*
8	*Support Interface*	● UART ● SPI ● I2C ● PWM
9	*Input power*	3.3V / 5V
10	Penggunaan Daya	● *Deep-sleep*: 6mA@5V ● *Modem-sleep*: 20mA@5V ● *Light-sleep*: 6.7mA@5V
11	*SPI Flash*	*Default* 32 Mbit

12	Berat	10 Gram

2. Kamera OV2640

Tabel 2.3 Spesifikasi Kamera OV2640

No.	**Spesifikasi**	**Nilai**
1	Resolusi	2 *Megapixels*
2	*Array size*	UXGA (1600 x 1200)
3	Lensa	1/4″ (6.35mm)
4	Maksimum *Image Transfer Rate*	15 *Frame* / detik
5	Format *Output* (8-bit)	• YUV (422/420)/YCbCr422 • RGB565/555 • 8-bit *compressed data* • 8-/10-bit *Raw RGB data*

2.5.2 ESP32-CAM CH340 *Development Board*

Gambar 2.13 ESP32-CAM CH340 *Development Board*

ESP32-CAM CH340 *Development Board* adalah *tools* untuk pengembangan proyek IoT dan pemrosesan citra yang utamanya digunakan untuk ESP32-CAM. Dengan *user interface* yang intuitif, tombol *reset* dan *flash*, indikator LED yang terintegrasi, serta manajemen daya yang kuat, *development*

board ini menyederhanakan proses penggunaan modul ESP32-CAM. Spesifikasi dari ESP32-CAM-CH340 *Development Board* dapat dilihat pada tabel 2.4.

Tabel 2.4 Spesifikasi ESP32-CAM CH340 *Development Board*

No.	Spesifikasi	Nilai
1	Model Produk	HW-818
2	Ukuran	27mm x 48.5 mm x 4.5 mm / 1.06” x 1.9” x 0.17”
3	*WI-FI*	802.11 b/g/n/e/i

2.6 Pelatihan Model

Bahasa pemrograman dan *framework* atau *library* yang digunakan adalah Python, Pytorch dan Matplotlib.

2.6.1 Python

Bahasa pemrograman Python adalah bahasa pemrograman tingkat tinggi yang sangat populer dan banyak digunakan dalam berbagai bidang. Diciptakan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991, Python dirancang dengan tujuan untuk menjadi bahasa pemrograman yang mudah dipahami, keterbacaan kode yang baik, dan sederhana dalam sintaksisnya (Baktikominfo, 2019).

Python menjadi bahasa pemrograman yang sangat populer dalam bidang data, terutama dalam *machine learning* dan *deep learning*. Keberhasilan Python dalam hal ini dapat dijelaskan oleh beberapa faktor utama:

1. Python memiliki ekosistem *library* dan *framework* yang kaya, seperti NumPy, Pandas, Matplotlib, dan TensorFlow, yang sangat mendukung pemrosesan data dan pengembangan model *machine learning*.
2. Kedua, Python menonjol dengan sintaksis yang sederhana, mudah dibaca, dan dapat dipahami oleh pemula maupun ahli, yang memudahkan para *developer* untuk memahami dan mengedit kode dengan cepat.
3. Ketiga, komunitas Python yang aktif memberikan dukungan dan *source code* yang berharga untuk para pengguna dan *developer*.

4. Terakhir, fokus pada produktivitas Python mempercepat siklus pengembangan, yang sangat penting dalam eksperimen dan iterasi cepat dalam bidang data.

Secara keseluruhan, Python telah terbukti sebagai bahasa yang kuat dan efisien dalam pemrosesan dan analisis data, dan menjadi pilihan utama bagi *data scientist* dan *developer* dalam menciptakan solusi inovatif dan efektif untuk tantangan dalam bidang data.

2.6.2 Pytorch

PyTorch adalah sebuah *framework open-source* yang digunakan dalam *Artificial Intelligence* (AI) dan *Deep Learning* (Yasar, 2023). *Framework* Pytorch dikembangkan oleh Facebook's AI Research lab (FAIR). Pytorch berfokus pada pembangunan dan pelatihan model *neural network*s dengan menggunakan bahasa pemrograman Python. Salah satu karakteristik utamanya adalah pendekatannya yang menggunakan graf komputasi dinamis, memungkinkan model dibangun secara dinamis selama waktu eksekusi program untuk fleksibilitas dalam mengatur aliran data dan struktur model.

PyTorch mendukung penggunaan GPU, yang mengizinkan percepatan paralel dalam pelatihan model. Kelebihan dalam keterbacaan dan kemudahan penggunaan membuat PyTorch menjadi pilihan favorit dalam kalangan peneliti, praktisi, *developer* dan mahasiswa di bidang *machine learning* dan *deep learning*.

2.6.3 Matplotlib

Matplotlib adalah sebuah *library* pada bahasa pemrograman Python yang digunakan untuk membuat visualisasi data secara statis, animasi, dan interaktif. *Library* ini memungkinkan pengguna untuk membuat plot berkualitas publikasi, membuat gambar interaktif yang dapat di-*zoom*, di-*pan*, dan di-*update*, menyesuaikan gaya visual dan tata letak dan mengekspor ke banyak format file. Matplotlib juga menyediakan antarmuka berbasis *state* (pyplot) dan berbasis objek yang lebih direkomendasikan untuk plot yang lebih kompleks.

2.7 Evaluasi Kinerja Model

2.7.1 *Confusion Matrix*

Confusion matrix adalah matriks evaluasi yang digunakan dalam klasifikasi untuk menampilkan jumlah data yang diklasifikasikan dengan benar atau salah dalam setiap kategori kelas. Matriks ini membantu untuk menganalisis

secara lebih rinci kinerja model dalam memprediksi kategori kelas yang benar atau salah berdasarkan data uji (Provost & Fawcett, 2013).

Dalam *confusion matrix*, terdapat empat komponen utama yang mewakili hasil dari proses klasifikasi, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) yang diilustrasikan pada tabel 2.5.

Tabel 2.5 Penjelasan *Confusion Matrix*

	Prediksi	
Sebenarnya	Positif	Negatif
Positif	*True Positive* (TP)	*False Negative* (FN)
Negatif	*False Positive* (FP)	*True Negative* (TN)

Berikut penjelasan dari 4 nilai dari tabel *confusion matrix*:

- *True Positive* (TP): Merupakan hasil klasifikasi yang menunjukkan bahwa model dengan benar memprediksi kelas positif.
- *True Negative* (TN): Merupakan hasil klasifikasi yang menunjukkan bahwa model dengan benar memprediksi kelas negatif.
- *False Positive* (FP): Merupakan hasil klasifikasi yang menunjukkan bahwa model secara keliru memprediksi kelas positif padahal seharusnya negatif.
- *False Negative* (FN): Merupakan hasil klasifikasi yang menunjukkan bahwa model secara keliru memprediksi kelas negatif padahal seharusnya positif.

2.7.2 *Recall*

Recall adalah metrik evaluasi dalam pengenalan pola dan deteksi objek yang mengukur sejauh mana model berhasil mengidentifikasi semua *instance* positif yang sebenarnya ada dalam data, memberikan gambaran seberapa baik model dalam menemukan objek-objek yang ada. (Everingham et al., 2009).

Dalam konteks deteksi objek, *recall* mengukur berapa persen objek positif yang berhasil diidentifikasi oleh model dari keseluruhan objek positif yang sebenarnya ada. Secara matematis *recall* memiliki persamaan sebagai berikut:

$$Recall = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)}$$

2.7.3 *Precision*

Precision adalah metrik evaluasi yang mengukur sejauh mana model benar dalam mengidentifikasi objek positif dari semua objek yang diidentifikasi oleh model, membantu memahami seberapa akurat deteksi objek yang dilakukan oleh model (Davis & Goadrich, 2006).

Dalam deteksi objek, *precision* mengukur berapa persen objek yang diidentifikasi sebagai positif oleh model yang benar-benar objek positif. Secara matematis, kita dapat menghitung *precision* dengan persamaan sebagai berikut:

$$Precision = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Positive(FP)}$$

2.7.4 *Mean Average Precision*

Mean Average Precision (mAP) adalah metrik evaluasi yang mengukur kualitas dan akurasi dari suatu model deteksi objek dalam menemukan dan mengklasifikasikan objek di dalam gambar atau video dengan menggabungkan *precision* pada berbagai tingkat *recall*. (Everingham et al., 2009). MAP umumnya digunakan dalam bidang deteksi objek dan pengenalan pola.

MAP menggabungkan dua konsep: *precision* dan *recall*. *Precision* mengukur seberapa akurat model dalam mengidentifikasi objek-objek positif (*true positive*) dari semua objek yang diidentifikasi (*true positive* + *false positive*). *Recall* mengukur seberapa banyak objek positif yang berhasil diidentifikasi oleh model dari semua objek positif yang sebenarnya ada (*true positive* + *false negative*). Secara matematis, mAP memiliki persamaan seperti berikut:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$

Untuk setiap kelas objek yang ada, hitung *Average Precision* (AP). AP dihitung dengan menghitung *precision* pada setiap tingkat *recall* dan kemudian mengambil rata-rata dari *precision* ini. Setelah mendapatkan AP untuk setiap kelas, hitung rata-rata dari semua AP ini untuk mendapatkan mAP. Perhitungan mAP akan memberikan gambaran tentang sejauh mana model mampu mengidentifikasi objek dengan baik pada berbagai tingkat *recall*.

2.7.5 *F1-Score*

F1-score adalah metrik evaluasi yang mengukur keseimbangan antara *precision* dan *recall*. *F1-score* didefinisikan sebagai rata-rata harmonik dari *precision* dan *recall* (Chicco & Jurman, 2020). *F1-score* memberikan gambaran holistik tentang kinerja model deteksi objek atau klasifikasi. Metrik *F1-Score* berguna terutama pada kasus di mana *precision* dan *recall* memiliki *trade-off* yang signifikan. *F1-score* dihitung dengan menggunakan rumus:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

2.7.6 *Loss*

Loss dalam konteks *machine learning* adalah metrik yang mengukur seberapa besar perbedaan antara prediksi yang dihasilkan oleh model dan nilai sebenarnya (*ground truth*) dari data pelatihan. Tujuan dari *loss* adalah untuk memberikan gambaran tentang sejauh mana model kita menghasilkan prediksi yang akurat dan sejalan dengan data pelatihan.

Loss function adalah faktor kunci dalam proses pelatihan model, karena membantu mengoptimalkan parameter-parameter model agar prediksi semakin mendekati nilai sebenarnya pada data pelatihan. Dalam berbagai aplikasi *machine learning*, seperti klasifikasi atau deteksi objek, berbagai jenis *loss function* digunakan untuk mengukur kinerja model dengan cara yang sesuai dengan tujuan spesifik tugas tersebut.

Dalam algoritma YOLO, terdapat tiga komponen *loss* utama yang digunakan untuk melatih model deteksi objek, yaitu *object loss, classification loss*, dan *box loss*. Ketiga *loss* ini memainkan peran penting dalam mengoptimalkan kemampuan model untuk mendeteksi objek secara akurat (Redmon et al., 2016). Selain ketiga komponen *loss* tersebut, terdapat satu tambahan *loss* yang ditambahkan pada YOLOv8, yaitu *distribution focal loss* atau *DFL Loss.*

a. *Object Loss* mengukur seberapa baik model dapat membedakan antara area dengan objek dan area tanpa objek dalam gambar. Ini membantu model untuk memahami dimana seharusnya mendeteksi objek dan di mana tidak.

b. *Classification Loss* mengukur seberapa akurat model dalam mengklasifikasikan objek yang telah terdeteksi. Ini membantu model memahami kelas objek yang sebenarnya dari gambar yang diberikan.
c. *Box Loss* mengukur seberapa akurat model dalam menghasilkan kotak pembatas (*bounding box*) yang tepat di sekitar objek yang terdeteksi. Ini membantu model memahami letak dan bentuk objek dengan lebih akurat.
d. *Distribution Focal Loss* dirancang untuk menangani ketidakseimbangan distribusi antar kelas objek dengan memberikan *loss*/penalti yang lebih besar pada kesalahan klasifikasi untuk kelas-kelas yang kurang umum. Dengan memperkenalkan faktor penyesuaian distribusi, *dfl loss* membantu model lebih fokus dan sensitif terhadap kelas-kelas yang sulit diidentifikasi, meningkatkan kemampuan deteksi objek dalam mengatasi variasi distribusi kelas yang tidak seimbang.

2.7.7 *Levenshtein Distance*

Levenshtein Distance adalah salah satu metrik yang umum digunakan dalam *natural language processing*, komputasi, ilmu komputer, dan bioinformatika. Metrik ini ditemukan oleh Vladimir Levenshtein pada tahun 1965 dan memiliki banyak aplikasi dalam berbagai domain salah satunya untuk permasalahan *string matching.*

Levenshtein distance adalah sebuah metrik *string* yang digunakan untuk mengukur perbedaan atau jarak (*distance*) antara dua *string*. Nilai *distance* antara dua *string* ini ditentukan oleh jumlah minimum dari operasi-perubahan yang diperlukan untuk melakukan transformasi dari suatu *string* menjadi *string* lainnya. Operasi-operasi tersebut meliputi penyisipan (*insertion*), penghapusan (*deletion*), atau penukaran (*substitution*) (Pratama & Pamungkas, 2016). Algoritma *Levenshtein distance* merupakan salah satu algoritma yang dapat digunakan dalam mendeteksi kemiripan antara dua *string* yang berpotensi melakukan tindak plagiarisme. Langkah-langkah algoritma *Levenshtein distance* dalam mendapatkan nilai *edit distance* meliputi:

1. Membuat matriks berukuran (m+1) x (n+1), di mana m dan n adalah panjang kedua *string* yang akan dibandingkan.

2. Menginisialisasi matriks dengan nilai 0 hingga m di baris pertama dan 0 hingga n di kolom pertama.
3. Menghitung nilai matriks dengan membandingkan setiap karakter dari kedua *string*. Jika karakter pada posisi yang sama sama, maka nilai matriks tetap. Jika tidak, nilai matriks diisi dengan nilai minimum dari (i-1, j) + 1, (i, j-1) + 1, atau (i-1, j-1) + 1, yang mewakili operasi penghapusan, penyisipan, atau penggantian.
4. Nilai *Levenshtein distance* adalah nilai di posisi (m, n) dari matriks tersebut.

2.8 Rancang Bangun *Website*

Berikut *framework* atau *tools* yang digunakan untuk membangun *website*:

2.8.1 ReactJs

ReactJS adalah *framework JavaScript* yang digunakan untuk membangun antarmuka pengguna (UI) pada aplikasi *web* (Balbier, 2023). ReactJs dikembangkan oleh Facebook dan pertama kali diperkenalkan pada tahun 2013. ReactJS dirancang untuk membantu *developer* dalam membangun aplikasi dengan antarmuka yang interaktif dan dinamis, terutama dalam pengembangan aplikasi satu halaman (*single-page applications*) yang memiliki banyak komponen yang perlu diperbarui secara efisien.

Reusability komponen, struktur pengembangan yang terorganisir, serta performa *render* yang cepat menjadikan ReactJS pilihan populer untuk proyek-proyek yang memerlukan antarmuka yang interaktif dan dinamis. Selain itu, dukungan untuk pengembangan aplikasi mobile melalui React Native juga memperluas cakupan penggunaan ReactJS.

2.8.2 FastAPI

FastAPI adalah sebuah *framework web* yang dikembangkan menggunakan bahasa pemrograman Python untuk membangun aplikasi *web* API dengan cepat dan efisien. *Framework* ini menggabungkan kecepatan eksekusi yang tinggi dengan kemudahan pengembangan dan dokumentasi otomatis.

FastAPI didesain dengan fokus pada kinerja yang optimal melalui penggunaan teknik seperti *asynchronous programming*, sehingga memungkinkan pengguna untuk menangani banyak permintaan sekaligus tanpa mengorbankan kecepatan. Selain itu, FastAPI menyediakan sistem validasi data yang kuat

menggunakan Pydantic, yang mempermudah validasi dan transformasi data masukan dan keluaran. *Framework* ini juga memiliki dukungan penuh terhadap *standar OpenAPI* dan *JSON Schema*, sehingga otomatis menghasilkan dokumentasi API yang rinci dan mudah dimengerti.

FastAPI populer di kalangan *developer* karena menggabungkan performa tinggi dengan pengembangan yang intuitif. Kemampuan *asynchronous programming*-nya memungkinkan aplikasi untuk menangani permintaan secara efisien dan tanggap, menjadikannya cocok untuk kasus penggunaan yang memerlukan waktu operasi yang singkat, seperti aplikasi *real-time* atau *streaming*.

2.8.3 MySQL

MySQL merupakan sebuah *relational database management system* (RDBMS) yang bersifat *open source* dan sering digunakan untuk mengelola basis data yang menggunakan bahasa SQL (Sitinjak et al., 2020). MySQL menyimpan data dalam bentuk tabel-tabel yang saling berelasi, dan memiliki kelebihan dalam kemudahan penyimpanan dan penampilan data. Selain itu, MySQL juga mendukung bahasa pemrograman PHP, dan merupakan salah satu aplikasi DBMS yang banyak digunakan oleh para pengembang aplikasi *web* (Tumini & Fitria, 2021). MySQL juga dikenal karena kehandalannya, selalu di-*update*, serta banyaknya forum yang memfasilitasi para pengguna jika mengalami kendala. MySQL juga memiliki kinerja yang cepat, handal, dan mudah digunakan, serta dapat bekerja dengan arsitektur *client-server* atau *embedded system* (Yuliansyah, 2014).

2.9 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah sebuah alat perancangan sistem yang berorientasi pada objek (Haviluddin, 2011). UML digunakan untuk membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek. UML terdiri dari berbagai jenis diagram, seperti *use case diagram*, *class diagram*, *sequence diagram*, dan *activity diagram*. UML dapat membantu *developer* sistem dalam merancang sistem yang akan dibuat karena dapat memudahkan dalam memodelkan sistem secara visual. UML juga dapat membantu dalam mengidentifikasi kebutuhan sistem, menggambarkan struktur sistem, dan menggambarkan interaksi antara komponen sistem (Widyatmoko & Pamungkas,

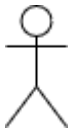
2022). UML banyak digunakan dalam pengembangan sistem informasi, seperti sistem informasi akademik, sistem informasi penggajian, dan sistem aplikasi pariwisata.

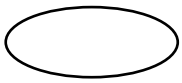
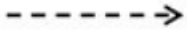
2.10 *Use Case Diagram*

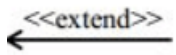
Use Case Diagram adalah sebuah jenis diagram pada UML yang digunakan untuk memodelkan perilaku suatu sistem dengan menggambarkan interaksi antara satu atau lebih aktor yang akan menggunakan sistem (Hutabri & Putri, 2019). Diagram ini bertujuan untuk mempresentasikan interaksi antara aktor dengan sistem, dimana aktor adalah entitas manusia atau sistem lain yang berinteraksi dengan sistem untuk melakukan pekerjaan tertentu.

Use Case Diagram menggambarkan fungsional yang diharapkan dari sebuah sistem, dan digunakan untuk mengetahui fungsi apa saja yang ada pada sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut (Simatupang & Sianturi, 2019). Diagram ini merupakan bahasa pemodelan perangkat lunak yang digunakan untuk perancangan sistem yang berorientasi objek. Use Case Diagram juga dapat membantu dalam mengidentifikasi kebutuhan sistem, menggambarkan struktur sistem, dan menggambarkan interaksi antara komponen sistem.

Tabel 2.6 Penjelasan *Use Case Diagram*

Simbol	**Nama**	**Keterangan**
	Aktor	Aktor adalah entitas di luar sistem yang berinteraksi dengan sistem. Aktor dapat berupa individu, kelompok, atau entitas lainnya yang berperan sebagai pengguna atau entitas eksternal yang berkomunikasi dengan sistem untuk mencapai

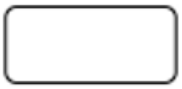
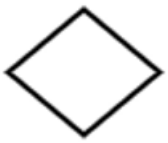
		suatu tujuan. Aktor tidak diperinci secara internal; fokusnya adalah pada peran atau fungsi mereka dalam berinteraksi dengan sistem.
	*Use Case*	*Use Case* merupakan representasi dari fungsionalitas atau tindakan yang dapat dilakukan oleh sistem. *Use case* menggambarkan cara sistem berinteraksi dengan aktor (entitas di luar sistem) untuk mencapai suatu tujuan tertentu.
	*Association*	*Association* merujuk pada hubungan antara dua elemen, seperti hubungan antara aktor dan use case
	Generalisasi	Generalisasi mengacu pada hubungan hirarkis antara *use case*. Ini menciptakan hubungan "*is-a*" antara *use case* yang lebih umum (*superclass*) dan *use*

		case yang lebih khusus (*subclass*). Dengan kata lain, *use case* yang lebih khusus mewarisi sifat dan perilaku dari *use case* yang lebih umum.
	*Extend*	Extend menunjukkan bahwa suatu *use case* (yang disebut *use cas*e dasar) dapat diperluas atau ditambahkan fungsionalitasnya oleh use case lain (yang disebut *use case* ekstensi).

2.10.1 *Activity Diagram*

Activity Diagram menggambarkan berbagai aliran aktivitas dalam sistem yang sedang dirancang, serta bagaimana masing-masing aliran berawal, keputusan yang mungkin terjadi, dan bagaimana aliran tersebut berakhir (Kurniawan & Syarifuddin, 2020). *Activity diagram* memodelkan aliran kerja atau urutan aktivitas dalam suatu proses yang mengacu pada *use case diagram* yang ada. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Simbol-simbol yang terdapat dalam *activity diagram* antara lain status awal/*initial state*, status akhir/*final state*, aktivitas, percabangan/*decision*, penggabungan/*join*, dan *swimlane* untuk memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. Beberapa simbol digunakan dalam pembuatan *activity diagram*, sebagaimana terlihat pada tabel 2.7.

Tabel 2.7 Penjelasan Simbol *Activity Diagram*

Simbol	**Nama**	**Keterangan**
	*Start* / Mulai	*Start* merepresentasikan titik awal atau permulaan dari suatu *activity* atau alur kerja. Simbol ini menandakan titik di mana suatu proses atau rangkaian *activity* dimulai dalam suatu sistem atau prosedur.
	*Activity*	*Activity* merepresentasikan suatu tindakan, *activity*, atau langkah yang terjadi dalam suatu proses. Simbol ini menunjukkan unit eksekusi dalam *activity* diagram dan seringkali mewakili suatu tugas atau aksi yang dilakukan oleh sistem, aktor, atau entitas lainnya dalam sistem.
	Percabangan / *Decision*	Percabangan/*Decision* digunakan untuk menunjukkan cabang atau pengambilan keputusan dalam alur

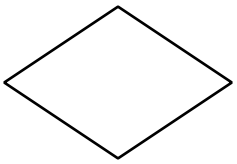
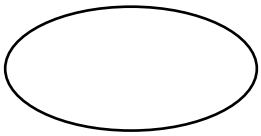
		kerja. Simbol ini mempresentasikan titik di mana sistem atau proses harus membuat keputusan berdasarkan kondisi tertentu. Keputusan ini kemudian mempengaruhi jalur mana yang akan diambil oleh alur kerja.
	Penggabungan / *Join*	*Join* digunakan untuk menunjukkan titik di mana dua atau lebih alur kerja atau jalur bergabung kembali setelah memisah pada suatu keputusan atau garis pemisahan. Simbol ini mengindikasikan bahwa dua atau lebih jalur eksekusi akan bergabung dan melanjutkan ke *activity* berikutnya setelah titik *join*

2.11 Metode Perancangan *Database*

Perancangan *database* adalah komponen integral dalam pengembangan sistem informasi. Salah satu pendekatan dalam merancang *database* adalah yaitu dengan menggunakan *Entity Relationship Diagram* (ERD). ERD adalah sebuah diagram struktural yang digunakan untuk merancang sebuah basis data (Latukolan

et al., 2019). ERD biasanya digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya dilakukan oleh sistem analis dalam tahap analisis persyaratan proyek pengembangan sistem. ERD akan mendeskripsikan data yang disimpan pada sebuah sistem maupun batasannya. ERD memiliki tiga konsep utama yaitu entitas, atribut, dan relasi. Adapun simbol-simbol yang digunakan pada pembuatan ERD dapat dilihat pada tabel 2.8.

Tabel 2.8 Penjelasan Simbol-simbol pada ERD

Simbol	Nama	Keterangan
	Persegi Panjang	Simbol persegi panjang digunakan untuk merepresentasikan entitas. Entitas dalam konteks *database* adalah objek yang menyimpan data dan memiliki keberadaan yang dapat diidentifikasi
	Belah Ketupat	Simbol belah ketupat digunakan untuk merepresentasikan hubungan (*relationship*) antara entitas. Hubungan ini menggambarkan bagaimana satu entitas terhubung atau berinteraksi dengan entitas lain
	Elips	Simbol elips (*ellipse*) digunakan untuk merepresentasikan atribut dari suatu entitas. Atribut adalah karakteristik atau properti yang mendefinisikan entitas dan menyimpan informasi tentang entitas tersebut.
	Garis	Simbol garis digunakan untuk merepresentasikan hubungan antara entitas, relasi dan atribut

2.12 Metode *Waterfall*

Waterfall adalah salah satu metode pengembangan perangkat lunak atau juga dikenal dengan istilah *Software Development Life Cycle. Waterfall* merupakan metode yang mengatur pengembangan perangkat lunak menjadi

serangkaian tahap, dan tiap tahap harus diselesaikan sebelum melanjutkan ke tahap berikutnya (Pressman, 2015). Metode *waterfall* membagi pengembangan menjadi beberapa tahap berbeda yang dilakukan secara berurutan:

1. *Requirement Analysis* (Analisis Kebutuhan)
2. *System and Software Design* (Desain sistem)
3. *Implementation* (Implementasi)
4. *Unit Testing* (Pengujian Sistem)
5. *Operation and Maintenance* (Operasi dan Pemeliharaan Sistem).

2.13 Penelitian Terkait

Perbandingan antara penelitian terkait dengan penelitian ini dapat ditemukan pada Tabel 2.9.

Tabel 2.9 Perbandingan Penelitian Terkait

No	Nama Peneliti	Judul & Tahun	Hasil Penelitian	Perbandingan Penelitian
1.	Dilshad Islam, Tanjim Mahmud & Tanjia Chowdhury	*An efficient automated vehicle license plate recognition system under image processing* (2022)	Peneliti berhasil merancang sistem pendeteksi dan rekognisi plat secara otomatis hanya dengan pemrosesan citra. Sebanyak 120 gambar plat kendaraan warna diuji dalam eksperimen ini, dan sistem ini memberikan. Akurasi 97,5% pada tahap ekstraksi plat kendaraan, 96,67% pada tahap segmentasi plat, dan 94,17% pada tahap rekognisi karakter plat kendaraan.	Dalam penelitian sebelumnya, sistem yang dibangun hanya berfokus pada rekognisi karakter pada plat kendaraan, dan sepenuhnya menggunakan teknik *image processing* tanpa keterlibatan *machine learning*. *Output* dari sistem berbentuk *file* notepad yang berisi hasil rekognisi plat kendaraan dari gambar yang di-*input*. Sedangkan dalam penelitian ini, fokus utama selain rekognisi karakter plat adalah penentuan ganjil-genap pada plat kendaraan dan patuh tidaknya kendaraan yang terdeteksi akan kebijakan ganjil genap. Sistem deteksi dibangun menggunakan YOLOv8 dan OCR sebagai dasar deteksi dan rekognisi plat kendaraan. *Input* gambar yang dideteksi merupakan

				gambar yang ditangkap langsung di lapangan secara *real-time*, dan *output* dari sistem ditampilkan pada *website*
2.	Reezky Illmawati & Hustinawati	YOLO v5 untuk Deteksi Nomor Kendaraan di DKI Jakarta (2023)	Peneliti berhasil mengimplementasi -kan sistem deteksi plat kendaraan menggunakan algoritma YOLOv5 dengan akurasi rata-rata pendeteksian objek sebesar 92.38%, rata-rata akurasi rekognisi karakter dalam plat nomor memiliki sebesar 95.45%, dan rata-rata akurasi pada tingkat keberhasilan program untuk menentukan kategori dari plat kendaraan sebesar 97.2%	Dalam penelitian sebelumnya, *input* pada model adalah 25 video yang telah direkam sebelumnya, dan model objek deteksi yang digunakan adalah YOLOv5. *Output* dari sistem berupa analisis akurasi dari 25 video *input*. Sedangkan dalam penelitian ini inputan berupa data yang direkam langsung oleh ESP32-CAM, dan model objek deteksi yang digunakan adalah YOLOv8. *Output* dari sistem pada penelitian ini berupa daftar pelanggar dan patuh yang telah terdeteksi dan ditampilkan pada *website*.
3.	Nurhaliza Juliyani Hayati	*Object Tracking* Menggunakan Algoritma *You Only Look Once* (YOLO) v8 Untuk Menghitung Kendaraan (2023)	Peneliti berhasil mengembangkan sistem penghitungan kendaraan secara otomatis menggunakan algoritma objek deteksi YOLOv8 dan algoritma pelacakan DeepSORT dengan akurasi sebesar 86%, presisi 86%, *recall* 86%, dan nilai *F1-Score* sebesar 85%.	Dalam penelitian sebelumnya, sistem yang dikembangkan bertujuan untuk otomasi penghitungan kendaraan menggunakan YOLOv8. Dalam penelitian ini, sistem yang dikembangkan berfokus untuk menegakkan kebijakan ganjil-genap plat kendaraan dengan cara mendeteksi dan rekognisi karakter plat kendaraan.
4.	Nauval Muhammad, Endro Ariyant	*Improved Face Detection Accuracy Using*	Peneliti berhasil merancang sistem *face recognition* berbasis ESP32-CAM yang	Tujuan yang ingin dicapai dalam penelitian sebelumnya berbeda dengan penelitian ini walaupun sama-sama menggunakan kamera ESP32-

	o & Yogi Anggun Saloko Yudo	*HAAR Cascade Classifier Method and ESP32-CAM For IOT-Based Home Door Security*	meningkatkan keamanan pintu rumah dengan akurasi rata-rata pada jarak 30 cm, 40 cm, dan 50 cm dengan intensitas cahaya 130 lux adalah 96,6%, serta dengan waktu rata-rata proses pengambilan sampel wajah hingga *face recognition* selama 21,50 detik	CAM sebagai perangkat untuk mengambil gambar atau merekam video. Penelitian ini berfokus dalam menerapkan sistem keamanan yang mempunyai aksi tergantung hasil klasifikasi wajah yang diberikan, misalnya wajah terdeteksi oleh sistem sebagai wajah yang terdaftar, maka pintu akan terbuka secara otomatis. Sedangkan penelitian ini berfokus pada sistem deteksi plat kendaraan. Pada penelitian ini, sistem tidak langsung memberikan aksi pada kendaraan yang terdeteksi melanggar atau patuh, sebaliknya sistem mencatat informasi-informasi tersebut dan menampilkannya di *website*.
5.	Victor Ciuntu & Hasan Ferdowsi	*Real-Time Traffic Sign Detection and Classification Using Machine Learning and Optical Character Recognition*	Peneliti berhasil merancang sebuah sistem kompleks deteksi dan klasifikasi rambu lalu lintas secara *real-time* dengan algoritma CNN dan OCR. Sistem yang diusulkan berhasil mendeteksi 92,8% rambu lalu lintas dengan benar dengan rata-rata waktu pemrosesan sebesar 251.527 detik. 7,2% dari rambu lalu lintas tidak terdeteksi, 10,1% rambu lalu lintas tidak terdeteksi secara benar, serta 90,76% akurasi dan 9,24% error pada	Walaupun sama-sama menggunakan OCR sebagai algoritma rekognisi karakter, penelitian sebelumnya menggunakan CNN sebagai algoritma untuk mendeteksi dan mengklasifikasikan objek rambu lalu-lintas pada Negara Belgia, Jerman dan Negara Bagian Illinois di USA. Pada penelitian ini, algoritma deteksi objek yang digunakan adalah YOLOv8. Penelitian ini bertujuan untuk mendeteksi dan merekognisi karakter plat kendaraan serta penegakan kebijakan ganjil-genap plat kendaraan pada Negara Indonesia.

			klasifikasi jenis rambu lalu lintas.	