BAB II
TINJAUAN PUSTAKA

2.1 Regulasi Pelanggaran Lalu Lintas

Di Indonesia, regulasi mengenai pelanggaran lalu lintas ditetapkan melalui Undang-Undang Republik Indonesia Nomor 22 Tahun 2009 tentang Lalu Lintas dan Angkutan Jalan. UU ini mencakup berbagai elemen, termasuk pentingnya lalu lintas dan angkutan jalan, pengembangan potensinya, serta peranannya dalam memastikan keamanan, keselamatan, ketertiban, dan kelancaran lalu lintas, serta akuntabilitas dalam penyelenggaraan negara [5].

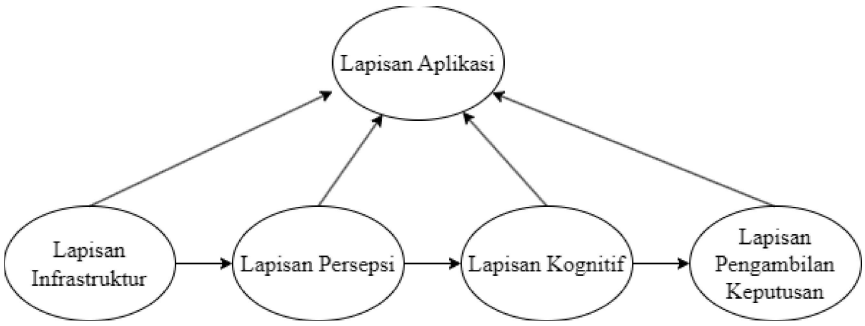
Salah satu poin penting dalam regulasi ini adalah kewajiban penggunaan helm bagi pengendara dan penumpang sepeda motor. Helm berfungsi sebagai perlengkapan keselamatan yang dirancang untuk melindungi kepala dari cedera serius saat terjadi kecelakaan. Aturan ini diatur secara spesifik dalam Pasal 57 ayat (1) dan ayat (2) yang menyatakan bahwa pengendara dan penumpang sepeda motor wajib menggunakan helm yang memenuhi standar nasional Indonesia [5].

Meskipun aturan ini telah diberlakukan sejak lama, tingkat kepatuhan pengendara masih beragam. Banyak pengendara belum menyadari pentingnya penggunaan helm, terutama di daerah-daerah terpencil. Kurangnya pengawasan dan penegakan hukum yang konsisten turut memperburuk situasi ini. Untuk itu, pemerintah Indonesia terus mencari cara untuk meningkatkan kesadaran dan kepatuhan terhadap aturan ini, salah satunya melalui penerapan teknologi dalam sistem pengawasan dan penegakan hukum lalu lintas, yaitu *Electronic Traffic Law Enforcement* (E-TLE). Dalam E-TLE, teknologi *Artificial Intelligence* diimplementasikan pada CCTV untuk mendeteksi objek dalam gambar atau video. E-TLE bekerja dengan menggunakan kamera pengawas yang ditempatkan di lokasi-lokasi strategis untuk memantau lalu lintas dan merekam pelanggaran, seperti melanggar lampu merah, melanggar garis berhenti, dan tidak menggunakan helm [8].

2.2 *Artificial Intelligence*

Artificial Intelligence (AI) atau kecerdasan buatan merujuk pada simulasi kecerdasan manusia dalam sistem berbasis digital, di mana tugas yang

memerlukan kecerdasan manusia dapat dilakukan tanpa masukan kecerdasan manusia [18]. Tujuannya adalah mengembangkan teknologi yang mampu memikirkan, merasakan, dan bertindak seperti manusia, termasuk dalam aspek-aspek seperti persepsi, penalaran, pembelajaran, perencanaan, prediksi, dan lain-lain. Berikut kerangka umum kecerdasan buatan yang disajikan pada Gambar 2.1.



Gambar 2.1 Kerangka Umum *Artificial Intelligence*[19]

Pada Gambar 2.1, kerangka pengembangan AI terdiri dari beberapa lapisan. Lapisan infrastruktur mencakup data, penyimpanan, daya komputasi, algoritma, dan *framework* AI yang mendukung kemampuan persepsi dan kognitif. Lapisan persepsi memberikan kemampuan dasar dalam penglihatan dan pendengaran, memungkinkan mesin untuk "melihat" dan "mendengar". Lapisan kognitif menyediakan kemampuan induksi, penalaran, dan akuisisi pengetahuan. Lapisan pengambilan keputusan memungkinkan mesin membuat keputusan optimal, seperti perencanaan otomatis dan sistem pendukung keputusan. Kemampuan ini mendukung aplikasi AI dalam berbagai bidang seperti ilmu pengetahuan, manufaktur, kehidupan, tata kelola sosial, dan dunia maya, yang mempengaruhi pekerjaan dan gaya hidup [19].

2.3 *Machine Learning*

Machine Learning adalah sebuah cabang ilmu komputer yang bertujuan untuk mengembangkan algoritma atau model yang mampu belajar dari data, mengidentifikasi pola, dan membuat prediksi atau keputusan tanpa adanya pemrograman eksplisit [20]. *Machine Learning* bertujuan untuk memungkinkan mesin untuk belajar secara mandiri, mengidentifikasi pola dan membuat prediksi yang akurat dan dapat diandalkan.

Machine learning menggunakan pendekatan matematis dan statistik untuk mengidentifikasi pola dan relasi yang tidak terlihat secara langsung dalam data, juga untuk meningkatkan efisiensi dan ketepatan model pembelajaran. Beragam masalah dapat diselesaikan dengan menggunakan teknik pembelajaran mesin, termasuk klasifikasi, regresi, pengelompokan, rekomendasi, pengenalan suara, pengenalan gambar, analisis bahasa alami, dan lain-lain [20].

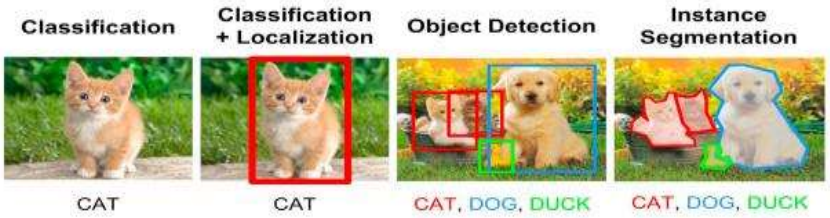
Machine learning dapat dibagi menjadi tiga jenis utama berdasarkan cara pembelajarannya, yaitu:

1. *Supervised learning*, yaitu pembelajaran dengan menggunakan data yang telah diberi label atau kelas. Tujuannya adalah untuk membuat model yang dapat memprediksi label atau kelas dari data baru yang belum diketahui sebelumnya. Contohnya adalah klasifikasi email menjadi spam atau tidak spam, klasifikasi objek pada gambar menjadi kategori tertentu, regresi harga rumah berdasarkan fitur-fiturnya, dan lain-lain [20].
2. *Unsupervised learning*, yaitu pembelajaran dengan menggunakan data yang tidak diberi label atau kelas. Tujuannya adalah untuk menemukan struktur atau kelompok dalam data tanpa adanya informasi sebelumnya. Contohnya adalah klastering pelanggan berdasarkan preferensi atau perilaku mereka, reduksi dimensi data untuk mempermudah visualisasi atau analisis, deteksi anomali atau outlier dalam data, dan lain-lain [20].
3. *Reinforcement learning*: yaitu pembelajaran dengan menggunakan umpan balik dari lingkungan sebagai penguat atau hukuman. Tujuannya adalah untuk membuat model yang dapat belajar dari pengalaman dan meningkatkan performa dalam mencapai tujuan. Contohnya adalah pelatihan robot untuk berjalan atau berlari, pelatihan agen untuk bermain game atau catur, pelatihan mobil otonom untuk mengemudi dengan aman, dan lain-lain [20].

2.4 Computer Vision

Computer vision adalah bidang ilmu yang berkembang pesat sejak beberapa dekade terakhir. Tujuan utama dari computer vision adalah untuk membuat komputer dan sistem mampu melihat, mengamati, dan memahami dunia

visual dengan cara yang mirip dengan manusia [21]. *Computer vision* awalnya berurusan dengan konsep-konsep seperti meniru sistem visual manusia melalui berbagai pemahaman tentang bagaimana skema kamera, proyeksi, dan fotogrametri bekerja. Dengan demikian, computer vision dapat dianggap sebagai cabang dari pengolahan citra, yang berfokus pada manipulasi dan analisis gambar digital [22]. Berikut contoh jenis sub-bidang pada *computer vision* yang diilustrasikan pada Gambar 2.2.



Gambar 2.2 Jenis Sub-bidang *Computer Vision* [23]

Seperti pada Gambar 2.2, *Computer vision* mencakup berbagai sub-bidang dan aplikasi yang berkaitan dengan dunia visual. Beberapa sub-bidang computer vision adalah rekonstruksi adegan, deteksi objek, pelacakan video, pengenalan objek, segmentasi objek, estimasi pose 3D, pembelajaran, pengindeksan, estimasi gerak, servo visual, pemodelan adegan 3D, dan restorasi gambar [23]. Beberapa aplikasi computer vision adalah deteksi wajah, pengenalan karakter optik, analisis medis, navigasi robot, pengawasan keamanan, dan sistem kendaraan pintar [22]. Pada penelitian ini akan berkaitan dengan penggunaan metode deteksi objek untuk perancangan sistem deteksi helm.

2.5 Deteksi Objek

Objek ditentukan oleh fitur seperti bentuk, ukuran, warna, tekstur, dan atribut lainnya. Deteksi objek tidak hanya menunjukkan keberadaan benda tetapi juga lokasinya dalam gambar. Proses ini melibatkan menemukan dan mengidentifikasi objek dunia nyata dalam gambar, serta menentukan posisinya menggunakan *bounding box*. Deteksi objek berkaitan erat dengan klasifikasi objek, segmentasi semantik, dan segmentasi instan[24][25].

Penerapan deteksi objek dapat dibagi menjadi berikut.

1. *Dedicated Object Detection*

Penerapan ini merujuk pada sistem deteksi objek yang dirancang untuk objek atau skenario tertentu, seperti deteksi pejalan kaki, deteksi kendaraan, atau mendeteksi sel kanker dalam gambar medis. Ini biasanya dilakukan dengan menggunakan model yang telah dioptimalkan atau dilatih khusus untuk tujuan tersebut, sehingga dapat memberikan hasil yang sangat spesifik dan optimal untuk jenis objek atau situasi yang ditargetkan [25].

2. *Generic Object Detection*

Pada penerapan deteksi objek ini, bertujuan untuk menemukan contoh objek dunia nyata dalam gambar tanpa terbatas pada kelas objek tertentu. Ini mencakup deteksi objek yang lebih umum seperti kendaraan, manusia, bangunan, dan lainnya. Dalam deteksi objek generik, model mungkin lebih umum dan tidak dioptimalkan secara khusus untuk satu jenis objek atau situasi tertentu, tetapi mampu mengenali berbagai jenis objek dengan tingkat akurasi yang cukup baik [25].

Dalam bidang deteksi objek, terdapat dua pendekatan utama yaitu sebagai berikut.

1. Metode Tradisional

Metode ini bergantung pada fitur yang dibuat secara manual dan algoritma untuk mendeteksi objek. Metode ini melibatkan langkah-langkah seperti pemilihan wilayah informatif, ekstraksi fitur, dan klasifikasi. Metode tradisional menggunakan teknik seperti deteksi tepi dan analisis tekstur untuk mengidentifikasi objek dalam gambar [26].

2. Metode *Deep Learning*

Metode ini menggunakan jaringan saraf untuk mempelajari fitur secara langsung dari data. Metode *deep learning* secara otomatis mengekstrak fitur hierarkis yang lebih tahan terhadap variasi dalam penampilan objek, skala, dan pose. Biasanya, mereka melibatkan pelatihan model pada dataset besar untuk mengenali pola dan membuat prediksi tentang lokasi dan kelas objek [26].

2.5.1 *Object Tracking*

Deteksi objek mengidentifikasi objek target dalam satu gambar, sementara pelacakan objek memprediksi posisi objek sepanjang rangkaian video.

Kemampuan ini penting untuk tugas visi komputer seperti pengawasan dan navigasi otonom. Pelacakan objek biasanya terjadi dalam dua konteks yang berbeda. Pelacakan objek tunggal (*Single Object Tracking*) berfokus pada satu target sepanjang video, contohnya adalah model yang berbasis *Siamese-Network* dan pelacak berbasis *Correlation Filter* [27]. Sedangkan pelacakan objek ganda (*Multiple Object Tracking*) atau pelacakan target ganda (*Multiple Target Tracking*) melibatkan pelacakan banyak objek secara bersamaan, model yang biasa digunakan misalnya SORT, DeepSORT, dan ByteTrack [28].

Dalam penelitian ini, digunakan pelacakan objek untuk proses *cropping* agar tidak terjadi pengulangan gambar yang di-*crop* pada setiap inferensi. Model ByteTrack digunakan karena ByteTrack mempertahankan kecepatan pemrosesan yang tinggi, melampaui SORT dan DeepSORT dalam hal efisiensi komputasi [28].

2.6 *Deep Learning*

Deep learning adalah cabang dari machine learning yang menggunakan jaringan saraf tiruan untuk menangani masalah kompleks dan membutuhkan banyak data. Jaringan ini memiliki banyak lapisan dan bobot sinapsis, yang memungkinkan menemukan pola atau hubungan tersembunyi antara *input* dan *output*. Hal ini membuat deep learning berbeda dari multilayer perceptron yang hanya memiliki tiga lapisan [29]. Dengan menggunakan banyak lapisan, *deep learning* meniru cara berpikir manusia dan dapat mengekstrak fitur tingkat tinggi dari data mentah seperti gambar, teks, suara, atau angka. Fitur-fitur ini kemudian digunakan untuk berbagai tugas seperti klasifikasi, regresi, klastering, dan generasi [30].

Pada kasus deteksi objek, penggunaan *deep learning* pada deteksi objek terbagi menjadi dua jenis sebagai berikut.

1. *One-stage Detector*

Pada jenis ini, tidak ada tugas perantara yang dilakukan dalam detektor satu tahap untuk menghasilkan output akhir. Hal ini membuat arsitekturnya lebih cepat dan sederhana. Sebagai contoh, YOLO dan SSD adalah detektor satu tahap

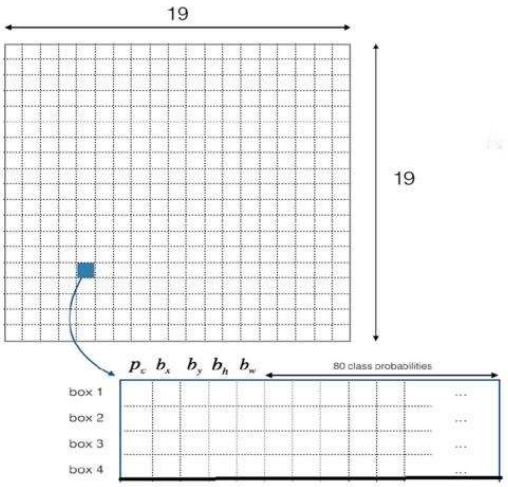
yang digunakan untuk menetapkan kotak pembatas pada posisi tertentu. Jadi, detektor ini melakukan pembelajaran pada lokasi objek tertentu [26].

2. *Two-stage Detector*

Metode ini menggunakan dua tahap untuk mendeteksi objek: pertama, menghasilkan kandidat area objek, kemudian membuat prediksi berdasarkan area tersebut. Di tahap pertama, detektor mengidentifikasi area semua objek. Detektor gambar menghasilkan area dengan tingkat *recall* tinggi, dan objek berada di setidaknya satu area ini. Pada tahap kedua, model *deep learning* melakukan klasifikasi. Area yang dihasilkan bisa berisi objek dengan label tertentu atau latar belakang. Model pada tahap pertama juga dapat memperbaiki lokalisasi[26].

2.7 YOLO

YOLO (*You Only Look Once*) merupakan suatu model deteksi objek yang memanfaatkan metode *one-stage* atau satu jaringan saraf konvolusi untuk memprediksi kotak pembatas dan kelas objek dalam sebuah gambar. YOLO hanya melihat gambar sekali (*You Only Look Once*), sehingga dapat melakukan deteksi objek secara *real-time*. Berbeda dengan model deteksi objek lainnya yang menggunakan dua jaringan saraf terpisah, YOLO dapat memperoleh hasil yang lebih cepat dan akurat karena memiliki arsitektur yang lebih efisien dan sederhana [31]. Untuk mendeteksi gambar dengan lebih cepat dan efisien, dilakukan pembagian grid gambar untuk melakukan deteksi seperti contoh pada Gambar 2.3.



Gambar 2.3 Pembagian gambar menjadi sel-sel grid dan prediksi yang sesuai untuk satu sel grid [31]

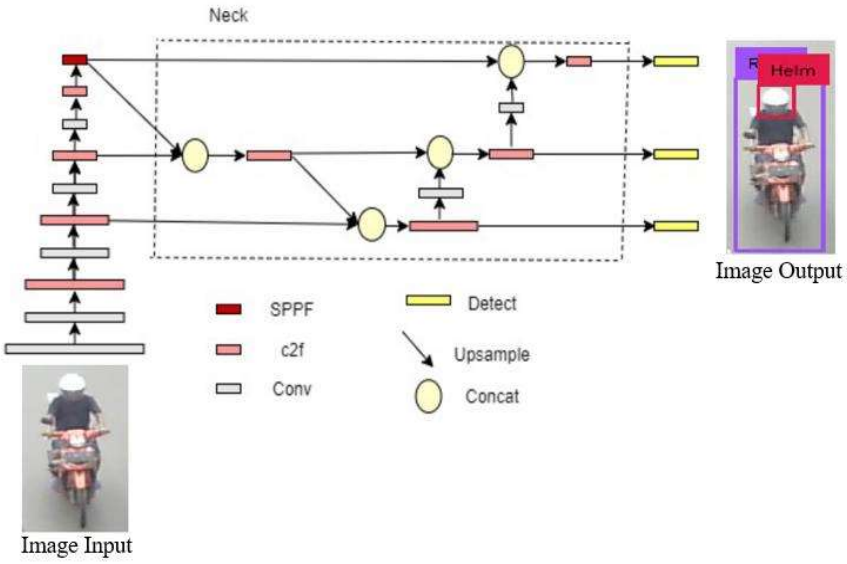
Pada Gambar 3.2, proses deteksi objek menggunakan YOLO membagi gambar menjadi *grid* berukuran $S \times S$, di mana setiap sel *grid* memprediksi B kotak pembatas, posisi dan dimensinya, probabilitas adanya objek, dan probabilitas kelasnya. Pusat objek harus berada dalam sel *grid* agar dapat terdeteksi. Setiap kotak pembatas diprediksi dengan parameter probabilitas objek (p_c), koordinat pusat kotak (b_x, b_y), dimensi kotak (b_h, b_w), dan probabilitas kelas (p_{ci}). Setiap sel *grid* memprediksi ($B \times 5 + n$) nilai, di mana B adalah jumlah kotak pembatas per sel dan n adalah jumlah kelas. Output tensor memiliki bentuk $S \times S \times (B \times 5 + n)$. Skor kepercayaan dihitung untuk setiap kotak pembatas dengan mengalikan p_c dengan *Intersection over Union* (IoU) antara kotak prediksi dan kotak asli. Jika tidak ada objek di sel *grid*, skor kepercayaannya adalah nol [31].

YOLO memiliki beberapa versi, seperti YOLOv1, YOLOv2, YOLOv3, YOLOv4, YOLOv5, YOLOv6, YOLOv7, dan YOLOv8. Setiap versi memiliki perbedaan dalam arsitektur, fitur, dan kinerja [32]. Pada penelitian kali ini, model YOLOv8 akan digunakan sebagai basis algoritma untuk sistem deteksi helm. Alasan pemilihan model ini akan dijelaskan lebih lanjut pada bagian teori yang menjelaskan tentang YOLOv8.

2.7.1 YOLOv8

YOLOv8 adalah algoritma deteksi objek dan segmentasi gambar terbaru dari Ultralytics, dirilis pada Januari 2023. Keunggulannya termasuk akurasi yang lebih baik dibandingkan model YOLO sebelumnya dan mendukung berbagai tugas seperti deteksi objek, instance segmentation, dan klasifikasi gambar. YOLOv8 fokus pada peningkatan akurasi dan efisiensi dengan arsitektur jaringan yang dioptimalkan, implementasi anchor box yang didesain ulang, dan fungsi loss yang dimodifikasi, menjadikannya lebih presisi dan fleksibel untuk berbagai aplikasi [33]. YOLOv8 menawarkan lima model berbeda yaitu N (*Nano*), S (*Small*), M (*Medium*), L *(Large*), dan X (*Extra Large*), masing-masing dengan kedalaman saluran dan jumlah filter yang bervariasi. Untuk arsitektur *backbone*

pada penelitian ini, dipilih versi *nano* dari YOLOv8 karena lebih ringan dibandingkan dengan versi lainnya [32]. Berikut arsitektur YOLOv8 yang digunakan pada penelitian kali ini, yang disajikan pada Gambar 2.4.



Gambar 2.4 Arsitekur YOLOv8 [34]

Pada Gambar 2.4 menjelaskan bahwa YOLOv8 memiliki struktur dasar yang terdiri dari beberapa lapisan, seperti lapisan konvolusi, lapisan C2F (*Cross Stage Partial Networks with Fusion*), dan lapisan SPPF (*Spatial Pyramid Pooling – Fast*). Lapisan C2F membantu mempercepat proses belajar dan konvergensi jaringan. Lapisan SPPF memungkinkan model mendeteksi objek dalam berbagai ukuran. Bagian leher (*neck*) menggabungkan dua jenis jaringan, yaitu FPN (*Feature Pyramid Network*) dan PAN (*Path Aggregation Network*), untuk membantu mengelola informasi dengan lebih baik. Fitur dari lapisan-lapisan yang berdekatan digabungkan dalam modul C2F. Jaringan ini menggabungkan fitur-fitur tingkat tinggi dengan fitur-fitur dasar saat melalui struktur tersebut. Bagian *output* memisahkan proses deteksi dan klasifikasi, menggunakan fitur tingkat bawah untuk mendeteksi objek kecil dan fitur tingkat atas untuk objek besar, dengan setiap lapisan deteksi menghasilkan informasi tentang lokasi dan kelas objek [34].

2.8 Teknik *Cropping* untuk Sistem

Teknik *cropping* pada sistem ini adalah komponen kunci dalam deteksi helm untuk memeriksa apakah dua objek tumpang tindih dalam suatu gambar. Dalam konteks ini, *overlap detection* digunakan untuk mendeteksi apakah ada pengendara yang terlihat tidak mengenakan helm. Deteksi ini dilakukan dengan memeriksa apakah *bounding box* 'Rider' tumpang tindih dengan *bounding box* 'Tidak Helm'. Dua *bounding box* dianggap tumpang tindih jika koordinat horizontal dan vertikalnya saling bertemu dalam area tertentu. Koordinat ini didapat dari *library* Ultralytics untuk YOLOv8 itu sendiri [35]. Secara matematis, kondisi tumpang tindih antara dua *bounding box* dapat dinyatakan pada Persamaan 2.1 dan Persamaan 2.2 sebagai berikut.

$$\max(x1_1, x2_2) < \min(x2_1, x2_2) \quad (2.1)$$

$$\max(y1_1, y2_2) < \min(y2_1, y2_2) \quad (2.2)$$

Fungsi ini, yang dijelaskan dalam Persamaan (2.1) dan Persamaan (2.2), memanfaatkan prinsip matematika untuk mengevaluasi apakah dua *bounding boxes* memiliki area yang tumpang tindih atau *overlap*. Persamaan (2.1) menguji posisi horizontal kedua kotak dengan membandingkan sisi kiri kotak yang posisinya lebih ke kanan (nilai x maksimum) terhadap sisi kanan kotak yang posisinya lebih ke kiri (nilai x minimum). Sementara itu, Persamaan (2.2) menilai posisi vertikal dengan membandingkan bagian atas kotak yang lebih rendah (nilai y maksimum) terhadap bagian bawah kotak yang lebih tinggi (nilai y minimum). Jika kedua kondisi ini dipenuhi, maka dapat disimpulkan bahwa kedua kotak tersebut memiliki irisan.

2.9 Metrik Evaluasi dalam *Computer Vision*

Dalam visi komputer, terdapat beberapa metrik evaluasi yang penting untuk mengukur kinerja dari suatu sistem atau model [23]. Berikut adalah beberapa metrik evaluasi yang digunakan pada penelitian.

1. Presisi (*Precision*): Presisi dalam deteksi objek mengukur kemampuan sistem untuk secara akurat mengidentifikasi objek yang terdeteksi. Dalam deteksi helm, tidak helm, dan pengendara, presisi tinggi menunjukkan

sedikitnya *false positive*, atau objek yang salah terdeteksi sebagai objek aktual.

2. Sensitivitas (*Recall*): Sensitivitas dalam deteksi objek mengukur seberapa baik sistem mendeteksi semua objek yang ada. Pada deteksi helm, tidak helm, dan pengendara, sensitivitas tinggi menunjukkan sistem dapat mendeteksi sebagian besar objek, mengurangi *false negative*.
3. F_1*-Score*: F_1*-Score* adalah rata-rata harmonis dari presisi dan sensitivitas. Dalam deteksi helm, tidak helm, dan pengendara, F_1*-Score* menunjukkan keseimbangan antara akurasi dan kemampuan deteksi. F_1*-Score* yang tinggi menandakan sistem memiliki presisi dan sensitivitas yang baik.
4. *Mean Average Precision* (mAP): mAP adalah nilai rata-rata presisi pada berbagai tingkat sensitivitas, memberikan gambaran kemampuan sistem mendeteksi objek dengan presisi yang konsisten. IoU mengevaluasi sejauh mana prediksi *bounding box* bertumpang tindih dengan objek sebenarnya, menjadi faktor kunci dalam perhitungan mAP. mAP50 dihitung dengan threshold IoU 0.5, sedangkan mAP50-95 memperhitungkan rentang IoU dari 0.5 hingga 0.95, memberikan gambaran lengkap tentang kualitas deteksi sistem pada berbagai tingkat kesulitan.

2.10 Jetson Nano

Jetson Nano adalah sebuah komputer *embedded* yang dikembangkan oleh NVIDIA. Jetson Nano dirancang untuk memberikan kemampuan komputasi yang tinggi dan efisien pada perangkat *edge* [36]. Jetson Nano menggunakan prosesor NVIDIA Maxwell dengan arsitektur GPU CUDA (*Compute Unified Device Architecture*), yang memungkinkan operasi komputasi paralel yang efisien. GPU dengan 128 inti CUDA dan CPU quad-core ARM Cortex-A57 memberikan daya komputasi yang handal [37]. Sistem operasi Jetson Nano adalah JetPak, yang merupakan versi khusus Linux yang telah dioptimalkan untuk komputasi AI. JetPak menyediakan lingkungan pengembangan yang lengkap dengan *library* dan *framework* AI yang sering digunakan seperti TensorFlow, PyTorch, dan Caffe [38].

Jetson Nano dapat digunakan dalam berbagai aplikasi AI, termasuk pengenalan objek, deteksi wajah, analisis citra, dan sebagainya secara *real-time*. Platform ini juga dilengkapi dengan berbagai antarmuka I/O seperti USB, HDMI, Ethernet, dan GPIO, yang memungkinkan integrasi dengan perangkat dan sensor eksternal [36]. Dengan performa tinggi dan fleksibilitas dalam integrasi perangkat, Jetson Nano dapat digunakan dalam berbagai proyek AI di perangkat edge seperti kendaraan otonom, robotika, sistem keamanan, dan IoT. Platform ini memberikan solusi yang efisien untuk pengembangan dan implementasi aplikasi AI yang membutuhkan daya komputasi tinggi dalam lingkungan *edge* [39].

2.11 Kajian Terdahulu

Dalam bagian ini dijelaskan beberapa penelitian yang terkait dengan topik penelitian tentang deteksi pelanggaran penggunaan helm pada pengendara motor. Beberapa studi ini melibatkan penggunaan model CNN seperti YOLO dan platform Jetson Nano untuk pengolahan data. Studi-studi ini dijadikan referensi dan landasan bagi penelitian yang sedang berlangsung.

Pada penelitian sebelumnya telah menjelaskan teknik-teknik yang digunakan untuk mengembangkan sistem deteksi pelanggaran penggunaan helm pada sepeda motor dengan menggunakan YOLO untuk mendeteksi sepeda motor, GoogleNet untuk mengklasifikasikan pelanggaran helm, dan *Kristan's method* untuk melacak objek [40]. Selain itu, penelitian ini juga mempersembahkan desain arsitektur sistem CPU-GPU berbiaya rendah namun berkinerja tinggi yang mampu memproses beberapa aliran kamera secara bersamaan. Hasil penelitian menunjukkan bahwa sistem CPU-GPU berbiaya rendah yang diusulkan mampu mendeteksi 97% pelanggaran penggunaan helm dengan tingkat kesalahan alarm sebesar 15%. Selain itu, model YOLO juga terbukti lebih efektif dalam mendeteksi objek pada sistem pelanggaran penggunaan helm pengendara motor dibandingkan dengan penggunaan HOG dan Haar Cascade dengan nilai *precision*, *recall*, dan *f1 score* yang lebih tinggi [40].

Penelitian selanjutnya yaitu mengenai sistem pendeteksi penggunaan masker dengan menggunakan Jetson Nano dan TensorFlow, serta teknik computer vision berbasis MobileNetV2 [41]. Sistem ini mencapai akurasi rata-rata 99,48%

dalam mendeteksi pengguna yang tidak mengenakan masker atau mengenakan masker dengan tidak benar, serta 99,12% dalam mendeteksi pengguna yang menggunakan masker dengan benar. Pada pengujian sistem ini, *platform* Jetson Nano memproses data gambar dalam waktu 0,114 detik, sebagai perbandingan, jauh lebih unggul daripada Raspberry Pi 4 yang memerlukan waktu kurang lebih 7 detik. Sehingga, dapat disimpulkan bahwa Jetson Nano memiliki performa yang lebih baik dalam memproses dan mengidentifikasi objek secara langsung [41].

Penelitian berikutnya membahas penggunaan *computer vision* dan *deep learning* untuk mendeteksi unta di jalan dan mencegah tabrakan antara kendaraan dan unta yang merupakan masalah serius di Arab Saudi dan daerah lainnya [42]. Penelitian ini membandingkan lima algoritma deteksi objek: CenterNet, EfficientDet, Faster R-CNN, SSD, dan YOLOv8, dengan menggunakan *dataset* khusus yang terdiri dari 250 gambar unta dalam berbagai konteks. Evaluasi dilakukan berdasarkan *mean Average Precision* (mAP) dan *mean Average Recall* (AR) pada berbagai threshold *intersection over union* (IoU). Hasil penelitian menunjukkan bahwa YOLOv8 adalah algoritma terbaik dari segi akurasi dan efisiensi, diikuti oleh CenterNet. Peneliti dari kajian ini menyimpulkan bahwa YOLOv8 adalah algoritma yang direkomendasikan untuk pengembangan sistem penghindaran tabrakan antara kendaraan dan unta berbasis *deep learning* di masa depan [42].

Selanjutnya pada penelitian ini menghadirkan sebuah proyek penelitian yang menggunakan YOLOv5 dan *ensemble learning* untuk secara otomatis mendeteksi dan mengklasifikasikan pengendara sepeda motor dan penumpangnya berdasarkan penggunaan helm [43]. Hasil menunjukkan skor mAP yang tinggi yaitu 0,526 pada data uji, dengan benar menandai sebagian besar kelas terlepas dari kondisi pencahayaan atau cuaca dari video. Meskipun demikian, model ini belum dapat diterapkan *deployment* secara *real-time* untuk menghadapi tantangan di dunia nyata seperti pengaruh perangkat *deployment* yang digunakan variasi kondisi pencahayaan, adanya halangan, dan berbagai jenis sepeda motor dan helm. Faktor-faktor ini dapat memengaruhikinerja dan keandalan model saat diterapkan dalam situasi praktis [43].