

BAB II

TINJAUAN PUSTAKA

2.1. Polresta Bandar Lampung

Polresta Bandar Lampung merupakan institusi kepolisian yang bertanggung jawab atas ketertiban dan keamanan, termasuk dalam aspek lalu lintas. Salah satu upaya modernisasi dalam pengelolaan lalu lintas di kota ini adalah penerapan Si-TIK (Sistem Informasi Teknologi Informasi Kepolisian), yang bertujuan untuk meningkatkan efisiensi penegakan hukum lalu lintas berbasis digital [6]. Selain itu, Polresta Bandar Lampung bekerja sama dengan Dinas Perhubungan dalam menerapkan sistem *Electronic Traffic Law Enforcement* (ETLE) guna menekan angka pelanggaran lalu lintas serta meningkatkan kepatuhan pengendara [7]. Satuan Lalu Lintas (Satlantas) memiliki tugas utama dalam pengaturan, penjagaan, pengawalan, dan patroli di jalan raya. Namun, tantangan utama yang dihadapi adalah kurangnya pemantauan lalu lintas secara *real-time* serta keterbatasan dalam mendeteksi titik-titik kemacetan yang dinamis (Mutiara, 2023)[8].

2.2. Lalu Lintas

Lalu lintas di Kota Bandar Lampung mengalami peningkatan yang signifikan dalam jumlah kendaraan yang beroperasi setiap tahunnya. Hal ini menyebabkan berbagai tantangan dalam pengelolaan lalu lintas, termasuk kepadatan kendaraan, kecelakaan, dan ketidakefisienan dalam distribusi arus lalu lintas [9]. Menurut penelitian yang dilakukan oleh Timotius (2023), faktor utama yang menyebabkan kemacetan di Bandar Lampung adalah pertumbuhan jumlah kendaraan yang tidak seimbang dengan peningkatan kapasitas jalan.

Penerapan sistem *Area Traffic Control System* (ATCS) di Bandar Lampung telah diuji coba sebagai solusi untuk mengatasi kemacetan, tetapi masih memiliki keterbatasan dalam hal respons terhadap kondisi lalu lintas yang cepat berubah (Aidila, 2023)[10]. Oleh karena itu, diperlukan inovasi teknologi yang lebih akurat dalam mendeteksi dan memprediksi kepadatan lalu lintas.

2.3. Manajemen Lalu Lintas

Manajemen lalu lintas bertujuan untuk mengoptimalkan penggunaan infrastruktur transportasi guna meningkatkan efisiensi dan keselamatan pengguna jalan. Beberapa strategi yang telah diterapkan di berbagai kota meliputi rekayasa lalu lintas, sistem kontrol lampu lalu lintas berbasis sensor, serta penerapan kebijakan pembatasan kendaraan [11].

Di Bandar Lampung, upaya penanggulangan kemacetan telah dilakukan melalui kebijakan pembatasan parkir di jalan utama serta peningkatan pengawasan oleh Dinas Perhubungan [12]. Namun, pendekatan konvensional yang masih dominan dalam manajemen lalu lintas menyebabkan kesulitan dalam mendeteksi pola kemacetan secara *real-time*. Oleh karena itu, penggabungan teknologi deteksi berbasis kecerdasan buatan (*AI*) seperti algoritma YOLO menjadi alternatif solusi yang dapat meningkatkan akurasi pemantauan serta efektivitas manajemen lalu lintas [13].

2.4. Algoritma YOLO (*You Only Look Once*)

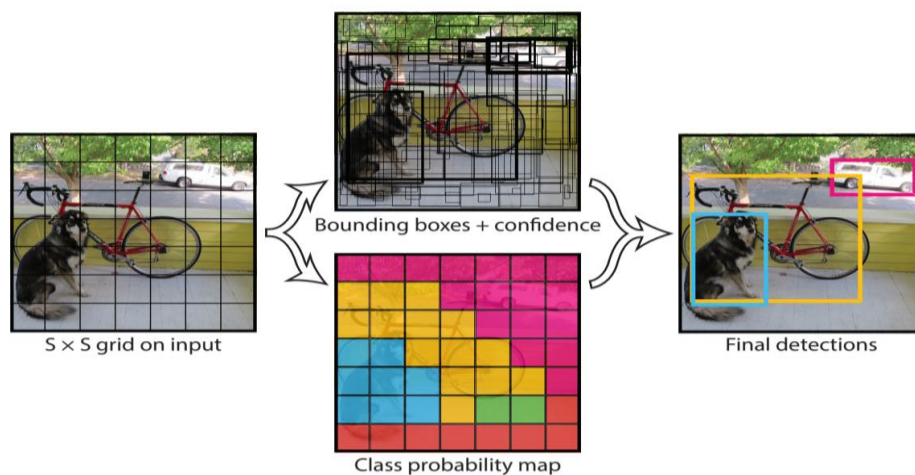
YOLO (*You Only Look Once*) adalah algoritma deteksi objek berbasis *deep learning* yang memungkinkan identifikasi dan klasifikasi objek dalam sebuah gambar atau video dalam waktu nyata. Dibandingkan dengan metode deteksi objek lainnya seperti R-CNN dan *Faster R-CNN*, YOLO lebih unggul dalam kecepatan deteksi karena menggunakan pendekatan *single-stage detection*, di mana seluruh gambar diproses dalam satu tahap tanpa perlu memotongnya menjadi beberapa bagian terlebih dahulu [14].

Algoritma YOLO telah banyak digunakan dalam berbagai aplikasi, termasuk pengawasan lalu lintas, pendeteksian kendaraan, dan sistem pengenalan wajah. Dalam konteks lalu lintas, YOLO dapat digunakan untuk mendeteksi kendaraan, mengidentifikasi jenis kendaraan, serta memprediksi tingkat kepadatan lalu lintas dengan akurasi tinggi [15].

Algoritma YOLO bekerja dengan membagi gambar *input* menjadi sejumlah grid dengan ukuran tertentu, misalnya $S \times S$ (seperti 13×13 pada YOLOv3). Setiap sel dalam *grid* memiliki tanggung jawab untuk mendeteksi objek yang pusatnya berada dalam sel tersebut. Setelah pembagian *grid* dilakukan, setiap sel akan memprediksi

sejumlah *bounding box*, yaitu kotak pembatas yang digunakan untuk menandai keberadaan objek, serta *confidence score* yang menunjukkan tingkat keyakinan model terhadap keberadaan objek dalam *bounding box* tersebut.

Selanjutnya, YOLO mengklasifikasikan objek yang telah terdeteksi dengan menggunakan jaringan saraf konvolusional (*Convolutional Neural Network* atau CNN). Namun, karena ada kemungkinan beberapa *bounding box* saling tumpang tindih untuk satu objek yang sama, maka digunakan metode *Non-Maximum Suppression (NMS)*. Teknik ini bertujuan untuk menyaring dan memilih *bounding box* terbaik dengan nilai *confidence score* tertinggi, sehingga duplikasi deteksi dapat dihindari. Pada tahap akhir, algoritma YOLO menghasilkan daftar *bounding box* yang telah dipilih, beserta label klasifikasi dan tingkat kepercayaan model terhadap hasil deteksi tersebut. Dengan metode ini, YOLO mampu mendeteksi objek dalam waktu nyata dengan kecepatan tinggi dan akurasi yang cukup tinggi dalam lingkungan yang kompleks, termasuk dalam skenario lalu lintas perkotaan [16].



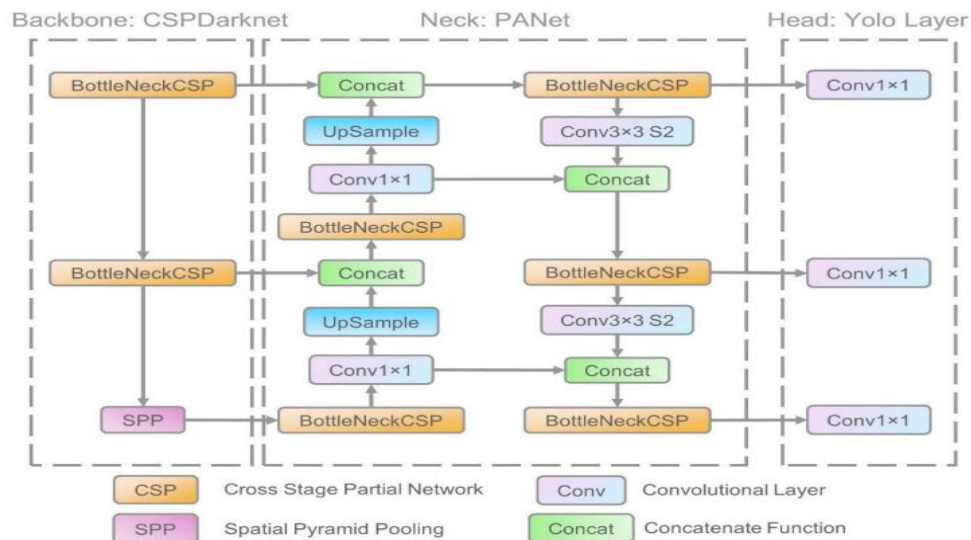
Gambar 2.1 Ilustrasi Cara Kerja YOLO

Arsitektur YOLO terdiri dari tiga komponen utama, yaitu *backbone*, *neck*, dan *head*. *Backbone* merupakan bagian pertama yang bertanggung jawab untuk mengekstraksi fitur dari gambar *input* menggunakan jaringan saraf konvolusional. Versi YOLO sebelumnya, seperti YOLOv3, menggunakan arsitektur Darknet-53

sebagai *backbone*, sedangkan YOLOv4 dan YOLOv5 telah mengadopsi CSPDarknet53 yang lebih efisien dan akurat dalam mendeteksi objek.

Setelah fitur diekstraksi oleh *backbone*, informasi tersebut diteruskan ke neck, yang bertugas untuk menggabungkan informasi dari berbagai skala menggunakan teknik *Feature Pyramid Network (FPN)* dan *Path Aggregation Network (PAN)*. Teknik ini memungkinkan YOLO untuk mendeteksi objek dalam berbagai ukuran dengan lebih baik.

Pada tahap akhir, *head* digunakan untuk melakukan prediksi *bounding box*, *confidence score*, serta klasifikasi objek dalam satu tahap pemrosesan. YOLO menggunakan metode *anchor boxes* untuk mengantisipasi ukuran objek yang bervariasi, sehingga deteksi dapat dilakukan lebih cepat dan akurat. Pada versi terbaru seperti YOLOv5 dan YOLOv7, arsitektur model telah dioptimalkan lebih lanjut dengan pendekatan *anchor-free detection* serta integrasi *EfficientNet* sebagai teknik ekstraksi fitur, yang meningkatkan kinerja YOLO dalam mendeteksi kendaraan di lingkungan lalu lintas yang kompleks dan dinamis [17].



Gambar 2.2 Arsitektur Jaringan YOLOv8

2.5. Deteksi Objek

Deteksi objek adalah salah satu bidang utama dalam visi komputer yang bertujuan untuk mengidentifikasi dan menemukan lokasi objek dalam suatu gambar atau video. Teknologi ini banyak diterapkan dalam berbagai bidang, seperti

keamanan, pemantauan lalu lintas, analisis citra medis, dan otomatisasi industri. Metode deteksi objek modern menggunakan pendekatan berbasis *deep learning*, seperti algoritma YOLO, *Faster R-CNN*, dan SSD (*Single Shot MultiBox Detector*) [18].

Dalam konteks lalu lintas, deteksi objek digunakan untuk mengenali kendaraan, pejalan kaki, dan rambu lalu lintas secara otomatis. Hal ini memungkinkan sistem manajemen lalu lintas untuk mengumpulkan data *real-time* yang dapat digunakan untuk analisis kepadatan kendaraan dan optimasi sinyal lalu lintas [19].

2.6. *Deep Learning*

Deep learning adalah cabang dari kecerdasan buatan yang menggunakan jaringan saraf tiruan berlapis-lapis (*deep neural networks*) untuk mempelajari pola dalam data. Metode ini sangat efektif dalam tugas-tugas kompleks seperti pengenalan gambar, pemrosesan bahasa alami, dan deteksi objek [20].

Dalam sistem deteksi lalu lintas, *deep learning* memungkinkan pengenalan pola kendaraan secara otomatis melalui analisis citra dan video. Algoritma seperti *Convolutional Neural Networks* (CNN) digunakan untuk mengekstrak fitur dari gambar kendaraan, sedangkan model seperti YOLO memungkinkan deteksi objek dalam waktu nyata dengan efisiensi tinggi [21].

2.7. *Google Colaboratory*

Google Colaboratory, atau yang sering disebut *Google Colab*, adalah platform berbasis *cloud* yang memungkinkan pengguna untuk menjalankan kode *Python* dalam lingkungan *Jupyter Notebook*. *Google Colab* menawarkan akses ke unit pemrosesan grafis (GPU) dan *tensor processing units* (TPU) secara gratis, sehingga sangat cocok untuk pengembangan model *deep learning* tanpa memerlukan perangkat keras yang mahal [22].

Dalam penelitian ini, *Google Colab* digunakan sebagai platform untuk melatih model YOLO dalam mendeteksi kepadatan lalu lintas. Dengan akses ke sumber daya komputasi yang kuat, proses pelatihan model dapat dilakukan dengan lebih cepat dan efisien.

2.8. *Roboflow*

Roboflow adalah platform berbasis *cloud* yang digunakan untuk memproses, mengelola, dan meningkatkan dataset gambar untuk keperluan visi komputer. *Roboflow* menyediakan berbagai fitur seperti augmentasi gambar, anotasi otomatis, serta integrasi dengan berbagai model *deep learning* seperti YOLO dan *Faster R-CNN* [23]. Dalam penelitian ini, *Roboflow* digunakan untuk meningkatkan kualitas dataset kendaraan yang akan digunakan dalam pelatihan model deteksi kepadatan lalu lintas.

2.9. *MongoDB*

MongoDB adalah sistem basis data *NoSQL* yang menggunakan model dokumen fleksibel untuk menyimpan data dalam format JSON. Keunggulan *MongoDB* dibandingkan basis data relasional adalah kemampuannya untuk menangani data tidak terstruktur dengan skala yang besar serta fleksibilitas dalam penyimpanan dan pengambilan data. Dalam konteks penelitian ini, *MongoDB* digunakan untuk menyimpan data hasil deteksi kepadatan lalu lintas yang dapat diakses secara *real-time* oleh sistem manajemen lalu lintas.

2.10. *Website*

Website adalah kumpulan halaman yang saling terhubung dan dapat diakses melalui internet menggunakan *Uniform Resource Locator* (URL). *Website* dapat berisi berbagai jenis informasi, mulai dari teks, gambar, video, hingga aplikasi interaktif yang memungkinkan pengguna untuk berinteraksi secara langsung dengan sistem. Berdasarkan sifatnya, *website* dapat dikategorikan menjadi *website* statis dan *website* dinamis.

Website statis memiliki konten yang tidak berubah kecuali diperbarui oleh pengembangnya secara manual. Sementara itu, *website* dinamis dapat menampilkan informasi yang diperbarui secara otomatis berdasarkan permintaan pengguna atau data yang tersimpan di dalam basis data. *Website* dinamis umumnya dikembangkan menggunakan bahasa pemrograman *server-side* seperti PHP, *Python* (*Django/Flask*), atau *Node.js*, serta basis data seperti *MySQL*, *PostgreSQL*, atau *MongoDB* untuk menyimpan dan mengelola data.

Pengembangan *website* dalam penelitian ini dilakukan menggunakan PHP sebagai bahasa pemrograman utama. PHP adalah bahasa pemrograman *server-side* yang umum digunakan dalam pembuatan *website* dinamis dan aplikasi web berbasis *database*.

Dalam sistem yang dikembangkan, *website* digunakan untuk menampilkan informasi kepadatan lalu lintas yang diperoleh dari model YOLO. *Website* ini memungkinkan pihak terkait, seperti dinas perhubungan dan kepolisian, untuk mengakses data kepadatan lalu lintas secara *real-time* dan mengambil keputusan yang tepat.

2.11. Metode Pengembangan Perangkat Lunak

2.11.1. Model Prototipe

Model prototipe adalah metode pengembangan perangkat lunak yang mengedepankan pembuatan versi awal sistem (prototipe) sebelum pengembangan penuh dilakukan. Pendekatan ini memungkinkan pengguna untuk mengevaluasi fitur utama perangkat lunak dan memberikan umpan balik untuk perbaikan sebelum sistem akhir dikembangkan [24].

2.11.2. Tahapan Model Prototipe

Model prototipe dalam pengembangan perangkat lunak terdiri dari beberapa tahapan utama yang bertujuan untuk memastikan bahwa sistem yang dikembangkan sesuai dengan kebutuhan pengguna.

1. Pengumpulan Kebutuhan

Tahap awal dalam model prototipe adalah mengidentifikasi kebutuhan pengguna serta spesifikasi awal sistem. Informasi ini diperoleh melalui wawancara, survei, atau analisis terhadap sistem yang sudah ada. Dalam konteks penelitian ini, pengumpulan kebutuhan dilakukan untuk memahami bagaimana sistem manajemen lalu lintas dapat dioptimalkan dengan algoritma YOLO.

2. Pembuatan Prototipe

Setelah kebutuhan dikumpulkan, tim pengembang membuat versi awal sistem yang mencakup fitur utama. Prototipe ini belum sepenuhnya fungsional, tetapi cukup untuk memberikan gambaran kepada pengguna mengenai bagaimana sistem akan bekerja. Pada penelitian ini, prototipe

berupa sistem deteksi kepadatan lalu lintas berbasis YOLO akan dikembangkan dan diuji secara awal.

3. Evaluasi Pengguna

Pengguna menguji prototipe yang telah dibuat dan memberikan umpan balik terkait fungsionalitas serta aspek lain seperti tampilan antarmuka dan kecepatan sistem. Evaluasi ini membantu mengidentifikasi kekurangan dan aspek yang perlu diperbaiki sebelum pengembangan lebih lanjut dilakukan.

4. Perbaikan dan Penyempurnaan

Berdasarkan umpan balik yang diberikan oleh pengguna, prototipe diperbaiki dan disempurnakan. Proses ini dapat berulang beberapa kali hingga sistem memenuhi ekspektasi pengguna. Dalam penelitian ini, perbaikan mungkin mencakup peningkatan akurasi deteksi kendaraan dan optimasi kecepatan pemrosesan data.

5. Pengembangan Produk Final

Setelah prototipe diuji dan diperbaiki sesuai dengan kebutuhan pengguna, pengembang melanjutkan ke tahap implementasi penuh. Pada tahap ini, sistem dikembangkan secara menyeluruh dengan semua fitur yang telah disempurnakan, diuji kembali, dan siap untuk digunakan dalam lingkungan nyata.

Pendekatan model prototipe ini sangat berguna dalam penelitian ini karena memungkinkan pengujian awal terhadap sistem deteksi kepadatan lalu lintas berbasis YOLO sebelum dikembangkan lebih lanjut menjadi sistem yang siap digunakan secara luas.

2.12. Penelitian Terkait

Berikut adalah tabel yang mendeskripsikan beberapa penelitian terdahulu yang memiliki keterkaitan dengan penelitian ini:

Tabel 2.1 Penelitian Terkait

Peneliti	Judul	Objek	Tahun	Metode	Hasil
Adri Surya Kusuma, Afu Ichsan Pradana, Bondan Wahyu Pamekas	Pengembangan Sistem Perhitungan Jumlah Kendaraan Berdasarkan Jenis Kendaraan Menggunakan Algoritma YOLO Secara <i>Realtime</i>	Kendaraan di jalan raya	2024	YOLOv9, pelacakan berbasis <i>centroid</i> , <i>confusion matrix</i>	Sistem mampu mendeteksi dan melacak kendaraan dengan akurasi 97,5% dalam klasifikasi dan 90% mAP dalam deteksi.
Muhammad Khairul Imam, Silvester Sari Sai, Adkah Yuliananda Mabur, Yosimson P. Manaha	Pemanfaatan Algoritma YOLO untuk Mendukung Sistem Informasi Lalu Lintas Berbasis <i>Geospatial Traffic Dashboard</i>	Kendaraan di Kota Mataram	2023	YOLO, CCTV, <i>geospatial dashboard</i>	Sistem dapat menghitung volume kendaraan secara otomatis dengan akurasi 93,387% dan menyajikan data lalu lintas dalam <i>dashboard geospasial</i> .
Jeremia Manurung, Nur Azizi, Disty Anastasya, Nicholas Valentino, Aditia Sanjaya, Kana Saputra	Deteksi Kemacetan Menggunakan YOLOv4 dan <i>Euclidean Distance Tracker</i> pada Jalan Raya	Kendaraan di Kota Medan	2023	YOLOv4, <i>Euclidean Distance Tracker</i>	Sistem mampu mendeteksi kendaraan dengan akurasi 61,3%, dan dengan integrasi <i>Euclidean Distance Tracker</i> , akurasi deteksi kemacetan mencapai 100%.
Muhamad Farhan Rajab, Fitrah Satrya Fajar Kusumah, Hersanto Fajri	Sistem Informasi untuk <i>Traffic Monitoring</i> di Kota Bogor Berbasis Web	Lalu lintas di Kota Bogor	2024	Metode <i>Waterfall</i> , <i>Codeigniter</i> , <i>Leaflet.js</i>	Sistem berhasil menyajikan informasi lalu lintas secara <i>real-time</i> , memudahkan pengguna mengetahui kondisi jalan dan angkutan umum.
Barokah Asmarahman Takarob	Deteksi dan Identifikasi Rambu-Rambu Lalu Lintas Berbasis Algoritma YOLO	Rambu lalu lintas	2022	YOLOv4, <i>OpenCV</i> , <i>Darknet</i>	Sistem mampu mendeteksi rambu dengan nilai mAP 100% dan akurasi 95,80% serta kecepatan pemrosesan 53,2 FPS.

Peneliti	Judul	Objek	Tahun	Metode	Hasil
I. Santoso & Y.A.A. Soetrisno	Perancangan Program Pendeteksi dan Pengklasifikasi Jenis Kendaraan Dengan Metode <i>Convolutional Neural Network (CNN) Deep Learning</i>	Deteksi kendaraan di lalu lintas	2020	<i>Convolutional Neural Network (CNN)</i>	Mendeteksi dan mengklasifikasikan kendaraan dengan akurasi tinggi menggunakan <i>deep learning</i> .
P. Harsadi & I. Ramadhan	Optimalisasi Sistem Parkir Menggunakan YOLO: Penerapan pada Lingkungan Urban Padat	Deteksi parkir di lingkungan padat	2024	YOLO untuk deteksi kendaraan	Sistem dapat mendeteksi kendaraan yang parkir dan mengoptimalkan penggunaan lahan parkir dengan YOLO.
R. Ramadan & H. Setiawan	Simulasi Pengontrol Lampu Lalu Lintas Berbasis Deteksi Kendaraan dengan YOLOv4	Sistem pengendalian lampu lalu lintas	2023	YOLOv4 dan <i>OpenCV</i>	Sistem mampu mengatur lampu lalu lintas berdasarkan kepadatan kendaraan yang terdeteksi.
PTD Indonesia-STTD	Implementasi Sistem Transportasi Cerdas	Sistem deteksi kecelakaan lalu lintas	2022	CCTV dan <i>AI</i>	Sistem dapat mendeteksi kecelakaan lalu lintas secara <i>real-time</i> menggunakan teknologi AI.
A.W. Wijayanto & I. Machdi	Implementasi Big Data dan Kecerdasan Artifisial untuk Statistik Oficial	Analisis kepadatan lalu lintas berbasis AI	2021	<i>Big Data & AI</i>	Sistem berbasis <i>big data</i> dapat digunakan untuk analisis statistik lalu lintas dan prediksi kepadatan di berbagai wilayah.