BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada saat ini, berbagai macam penelitian untuk pengembangan sistem pendukung *driver* terus berjalan mengacu pada aspek keselamatan pengendara yang dapat membantu pengendara dalam mengenali lingkungan sekitar untuk memitigasi potensi terjadinya kecelakaan. Bahkan, berdasarkan dari hasil studi oleh McKinsey & Co., bahwa *Advanced Driver Assistance System* (ADAS) dan *Autonomous Vehicle* (AV) dapat menekan angka kecelakaan sebesar 90% (Brown, 2015). Kendaraan tanpa awak atau *autonomous vehicle* diyakini mampu mengurangi kesalahan-kesalahan berkendara pada manusia. Namun, sistem tersebut memerlukan banyak fitur dalam mengenali lingkungan sekitar agar dapat dikatakan layak dan aman untuk digunakan. Dalam melakukan pengenalan environment sekitar, sistem ADAS menggunakan kamera stereo untuk menangkap gambar dari beberapa sudut. Terdapat suatu teknik yang dikenal sebagai *computer vision* yang mampu mengelola citra atau gambar yang diambil dari kamera untuk melakukan tugas pengenalan objek yang terdapat pada citra. Dengan teknik tersebut, input atau campur tangan manusia sebagai pemegang kendali pada kendaraan dapat diminimalisir.

Di Indonesia, *speed bump* dan *rumble strip* diajukan sebagai kebijakan dalam pengadaan alat pembatas kecepatan pengguna jalan untuk memperlambat kecepatan kendaraan demi keselamatan pengguna jalan berdasarkan Peraturan Menteri Perhubungan Republik Nomor 82 Tahun 2018 (*Permenhub No. 82 Tahun 2018*, 2018). Alat pembatas kecepatan pengguna jalan yang dimaksud diantaranya adalah *speed bump* dan *rumble strip* yang keduanya memiliki fungsi untuk mereduksi kecepatan kendaraan di jalanan. Kedua alat ini merupakan peninggian sebagian badan jalan untuk mengurangi kecepatan kendaraan. Berdasarkan penelitian oleh June M. Tester, *speed bump* sangat efektif dalam menekan kasus kecelakaan dengan penurunan 53% hingga 60% terhadap cedera dan korban jiwa pada anak akibat kecelakaan mobil (Tester dkk., 2004). Namun karena wujudnya yang menyerupai gundukan juga membuatnya merupakan *obstacle* yang perlu diantisipasi keberadaannya oleh manusia sehingga manusia mendapatkan pengalaman berkendara yang nyaman dan aman. Sepanjang tahun 2023, terdapat 148.307 kecelakaan lalu lintas di Indonesia (Media, 2024). Beberapa kecelakaan tersebut diantaranya diakibatkan oleh kedua objek tersebut. Dosen muda di Universitas Hasanuddin meninggal dunia akibat terjatuh dari kendaraannya setelah melintasi *speed bump* (Yunus, 2022). Selain itu terdapat *rumble strip* yang beberapa kali mengakibatkan kecelakaan tunggal di depan Mapolres Bondowoso (Adi, 2024).

Bila melihat data dari Pusiknas Polri tahun 2022, Faktor kurangnya konsentrasi pengendara sehingga tidak memerhatikan jalan menjadi faktor penyebab kecelakaan terbanyak dengan 18.070 kasus (Lestari, 2023). Kebiasaan buruk pada pengendara dan ditambah dengan wujud dari *speed bump* dan *rumble strip* yang dapat menjadi *obstacle* bagi pengguna jalan maka salah satu fitur yang dapat menjadi referensi pada sistem ADAS adalah pendeteksian *speed bump* dan *rumble strip* pada jalanan. Kehadiran fitur

tersebut pada sistem ADAS dapat membantu untuk mengetahui keberadaan objek tersebut lebih dini sehingga sistem pada *autonomous vehicle* mampu melakukan pengambilan keputusan untuk memberikan rasa nyaman bagi pengguna dan dapat meminimalisir bahaya-bahaya yang berpotensi ditimbulkan kedepannya.

1.2 Teori

1.2.1 *Speed Bump*

Sesuai aturan pada Peraturan Menteri Perhubungan Nomor PM 48 Tahun 2023, *speed bump* merupakan alat pembatas kecepatan yang digunakan hanya pada area parkir, jalan privat, atau jalan lingkungan terbatas dengan kecepatan operasional di bawah 10 (sepuluh) kilometer per jam (PUSDATIN, 2023). Spesifikasi teknis *speed bump* tidak dijelaskan lebih lanjut pada peraturan ini dan pada kenyataannya, sangat banyak masyarakat yang membangun *speed bump* atau yang biasa dikenal sebagai polisi tidur secara mandiri. Karakteristik yang sama antara polisi tidur yang dibuat oleh masyarakat yaitu berbentuk penampang melintang seperti yang ditunjukkan pada Gambar 1:

Sumber: (Aditya, 2018)

Gambar 1 Ilustrasi *Speed Bump* atau Polisi Tidur

1.2.2 *Rumble Strip*

Berdasarkan pada Peraturan Menteri Perhubungan Nomor PM 48 Tahun 2023, *Rumble strip* termasuk ke dalam jenis pita penggaduh yang merupakan kelengkapan pada jalan yang berfungsi sebagai (PUSDATIN, 2023):

Pertama. Mengurangi kecepatan kendaraan.
Kedua. Mengingatkan pengemudi tentang objek di depan yang harus diwaspadai.
Ketiga. Melindungi penyeberang jalan.
Keempat. Mengingatkan pengemudi akan lokasi rawan kecelakaan

Sama seperti *speed bump*, tidak disebutkan standar pembuatan *rumble strip* pada peraturan tersebut namun *rumble strip* biasanya dibangun pada jalan raya sebelum *zebra cross* ataupun tempat-tempat dimana biasanya pejalan kaki menyeberangi jalanan ataupun wilayah-wilayah yang menjadi lokasi rawannya kecelakaan kendaraan. *Rumble strip* memiliki bentuk yang cukup mirip dengan *speed bump* namun dengan ketinggian

yang lebih rendah dan di cat dengan warna putih dan dipasang berulang dengan jarak yang cukup dekat seperti yang ditunjukkan pada Gambar 2.

Sumber: (Aditya Pratama & Hartawan, 2020)

Gambar 2 Ilustrasi Rumble Strip atau Pita Penggaduh

1.2.3 *Computer Vision*

Computer Vision merupakan bidang keilmuan yang menggunakan teknologi komputer untuk memproses gambar sedemikian rupa sehingga mampu mengekstraksi informasi-informasi yang terdapat pada gambar. *Computer vision* mencoba untuk mengadopsi cara manusia dan hewan dalam merekonstruksi properti pada gambar seperti bentuk, iluminasi, dan penyebaran warna meskipun masih terdapat kemungkinan *error* atau salah persepsi (Szeliski, 2022).

Saat ini, penggunaan *computer vision* sudah banyak diterapkan pada beberapa bidang untuk membantu pekerjaan-pekerjaan manusia. *Computer vision* biasanya diterapkan untuk melakukan tugas-tugas seperti deteksi objek, pengenalan objek pada gambar, identifikasi objek pada gambar, rekonstruksi dari *view* tiga dimensi untuk menentukan posisi objek, dan pelacakan gerakan objek (Zayniddinov dkk., 2023).

Computer Vision bekerja dengan melalui beberapa tahapan. Tahapan-tahapan ini merupakan pendekatan dari cara mata manusia dan hewan dalam melihat. Tahapan dalam computer vision melalui pengambilan gambar, pemrosesan awal pada gambar, mengidentifikasi karakteristik atau fitur dari objek, deteksi atau segmentasi, pemrosesan tingkat tinggi (Zayniddinov dkk., 2023).

1.2.4 *Deep Learning*

Deep Learning adalah cabang bidang keilmuan dari *machine learning* yang mencoba untuk mempelajari abstraksi yang tinggi dengan memanfaatkan arsitektur hirarki. Arsitektur tersebut mampu melakukan ekstraksi fitur otomatis dari data mentah (Guo dkk., 2016). Model *deep learning* dibangun dengan jaringan syaraf tiruan. *Deep learning* memiliki lapisan tambahan yang disebut dengan *hidden layer*. *Hidden layer* ini yang berfungsi untuk mengekstraksi fitur. Fitur *Low-level* diekstraksi oleh layer terbawah sedangkan fitur *high-level* diekstraksi oleh layer teratas (Pouyanfar dkk., 2018). Sesuai dengan namanya, *deep learning* memiliki tingkat atau kedalaman yang lebih tinggi pada lapisan node atau unitnya yang sangat berbeda bila dibandingkan dengan model belajar yang lebih dangkal pada layer unitnya. Perbedaan ini memungkinkan *deep learning* untuk memetakan fungsi-fungsi non-linear dan kompleks, yang mana hal ini sangat tidak efisien bila dilakukan pada model belajar yang lebih dangkal arsitekturnya.

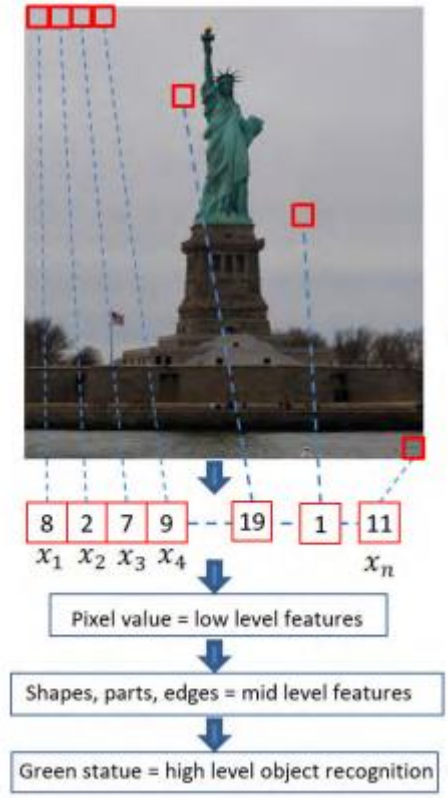
Hidden layer tersambung dengan *input layer*, *output layer*, atau *hidden layer* lainnya. Setiap layer memiliki node atau unit yang kemudian setiap node pada setiap layer ini terkoneksi dengan node pada layer tetangganya. Setiap koneksi ini memiliki nilai bobot masing-masing. Nilai Bobot ini digunakan untuk memperbarui input dan akan dijumlahkan pada setiap node atau unit (Shrestha & Mahmood, 2019). Pada *computer vision, deep learning* berperan pada 3 tahapan terakhir, yaitu:

Pertama. Mengidentifikasi karakteristik atau fitur dari objek.
Kedua. Deteksi atau segmentasi.
Ketiga. Pemrosesan tingkat tinggi.

Deep Learning sangat berguna untuk digunakan pada masalah-masalah yang berkaitan dengan dimensi data yang sangat besar. Dimensi data yang sangat besar dalam hal ini merupakan data data yang melibatkan fitur atau atribut yang sangat banyak yang bila direpresentasikan dengan vektor maka memerlukan ruang dimensi yang besar juga. Hal ini menjadi keunggulan dari *deep learning* dibanding dengan algoritma tradisional lainnya seperti *machine learning.* Dari kelebihan ini, *deep learning* sangat bermanfaat, berguna dan unggul pada bidang-bidang seperti pengenalan pola, *computer vision,* pemrosesan bahasa alami (NLP), pengenalan suara dan audio.

1.2.5 *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan salah satu arsitektur jaringan yang dikembangkan sebagai lapisan node atau unit pada *deep learning.* CNN dibangun untuk menyelesaikan tugas-tugas khususnya pada permasalahan di bidang yang berkaitan pada pengenalan gambar ataupun *computer vision.* CNN mampu untuk mengatasi tugas-tugas yang melibatkan data-data dengan informasi spasial di dalamnya. Arsitektur pada CNN memiliki hirarki lapisan untuk mempelajari fitur. Terdapat tahapan-tahapan yang perlu dilalui dengan fungsi yang berbeda pada arsitektur ini. Contohnya seperti pada Gambar 3.



Sumber: (Shrestha & Mahmood, 2019)

Gambar 3 Ilustrasi Ekstraksi Fitur pada CNN

Ketika melakukan tugas dalam pengenalan objek pada gambar, input gambar akan melalui lapisan awal yang bertugas untuk mengekstraksi fitur-fitur sederhana seperti nilai piksel. Kemudian memasuki layer-layer selanjutnya, fitur-fitur lain yang lebih kompleks akan diekstraksi secara perlahan seperti bentuk, tepian, dan sudut pada objek. Pada akhirnya, memasuki layer terakhir fitur yang akan diekstraksi merupakan fitur tingkat tertinggi yaitu objek yang terdapat pada gambar.

Pada *Convolutional Neural Network* (CNN), terdapat lapisan *convolutional* yang menerima input gambar berupa *array* 3 dimensi dengan ukuran seperti ditunjukkan pada persamaan (1):

$$w \times h \times r \quad (1)$$

Keterangan:

w : *width* (Lebar gambar)

h : *height* (Tinggi gambar)

r : *depth / channel* (r = 3, gambar berupa RGB)

Input ini akan digunakan untuk melakukan perkalian titik dengan nilai bobot pada *convolutional layer* yang kemudian keluaran dari *convolutional layer* diaplikasikan ke dalam fungsi non-linear atau fungsi aktivasi sesuai persamaan (2).

$$h^k = f(w^k * x + b^k) \quad (2)$$

Keterangan:

h^k : Keluaran dari *layer* ke-k
f : Fungsi aktivasi
w^k : Nilai beban
x : Vektor input
b^k : Nilai vektor bias

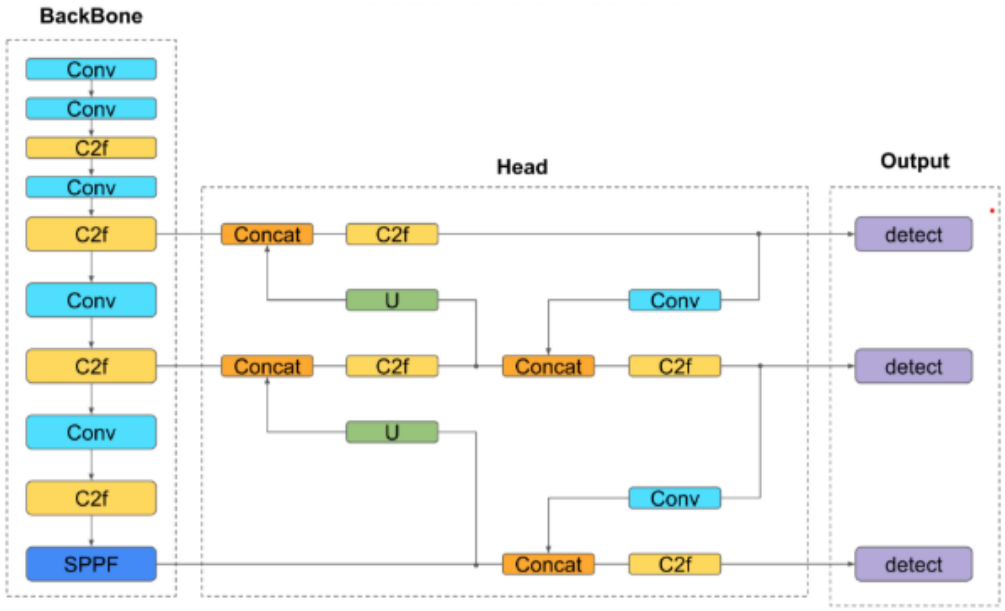
Selain itu terdapat beberapa lapisan lain yang setelah dari *convolutional layer*, diantaranya adalah:

Pooling (subsampling) layer. Pooling atau subsampling layer ini berfungsi untuk mengurangi jumlah fitur dan mereduksi dimensi dari keluaran convolutional layer sehingga mempercepat berjalannya proses pembelajaran. Proses pooling ini dijalankan dengan melewatkan $n \times n$ filter (n merupakan ukuran filter) di semua wilayah pada keluaran convolutional layer.

Fully Connected Layer. Lapisan ini mengambil fitur-fitur tingkat rendah yang kemudian akan menghasilkan fitur tingkat tinggi yaitu objek pada gambar. Hasil dari lapisan ini akan digunakan oleh lapisan akhir untuk melakukan proses klasifikasi dari objek yang ditemukan pada gambar.

1.2.6 You Only Look Once (YOLO)

You Only Look Once atau YOLO merupakan model deteksi objek yang dikenal dengan akurasi dan ukurannya yang cukup ringan. Versi terbaru dari jaringan YOLO pada saat ini yaitu YOLOv8 yang dikembangkan oleh Ultralytics sebagai pengembangan dari YOLOv5. YOLO mampu melakukan pendeteksian objek dengan hanya melalui 1 jaringan. Hal tersebut yang membedakannya dengan model-model sebelumnya yang menggunakan metode yang memerlukan tahapan pertama untuk mendeteksi wilayah berpotensi objek dan tahapan untuk melakukan klasifikasi pada usulan wilayah (Terven dkk., 2023). Jaringan dari model YOLOv8 pada dasarnya terdiri dari 3 bagian seperti diilustrasikan pada Gambar 4:



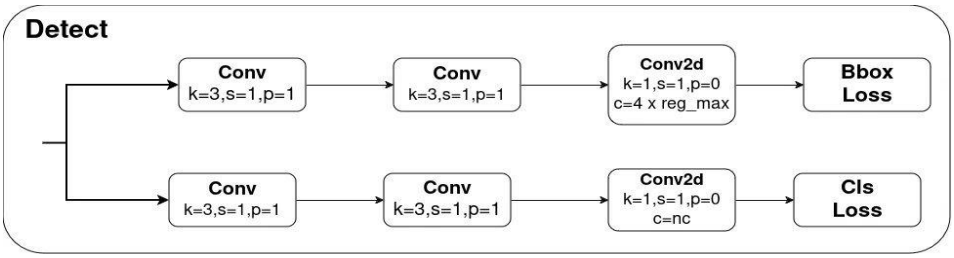
Sumber: (Sohan dkk., 2024)

Gambar 4 Arsitektur YoloV8

Backbone. *Backbone* bertugas sebagai ekstraksi fitur yang menggunakan beberapa *convolutional layer* dan menggunakan teknik *cross-partial stage connections* (C2f) yang mengkombinasikan fitur level tinggi dengan informasi tambahan yang berada di sekitar objek untuk meningkatkan akurasi deteksi. Terdapat 5 blok *Convolutional Layer* dan 4 blok C2F *Layer* dan 1 blok SPPF yang berguna untuk mengekstrak 3 ukuran fitur map untuk mendapatkan fitur objek pada variasi ukuran kecil, sedang, dan besar (Sohan dkk., 2024).

Head. Bagian *head* menerima peta fitur keluaran dari *backbone* dan kemudian dengan menggunakan rangkaian *convolutional layer* untuk menganalisa peta fitur dan menggunakan *linear layer* untuk melakukan prediksi beberapa *bounding box* dan nilai probabilitas atau skor kelasnya. Keduanya merupakan keluaran dari modul *head* pada arsitektur YOLOv8 (Sohan dkk., 2024).

Modul deteksi. Modul deteksi menggunakan keluaran dari modul *head* untuk memetakan fitur berdimensi tinggi terhadap keluaran beberapa *bounding box* dan kelas objek dengan menggunakan *convolutional layer* dan *linear layer* (Sohan dkk., 2024).



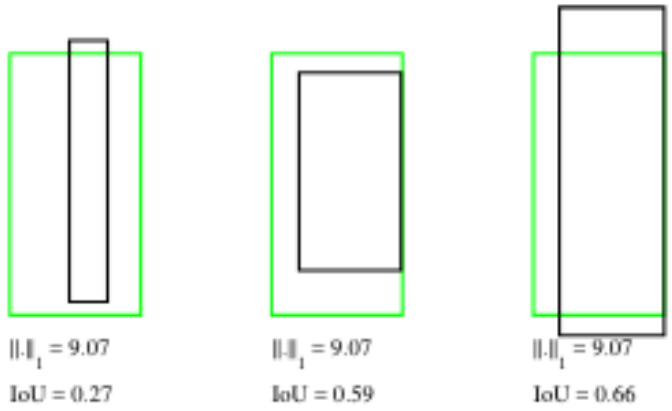
Sumber: (Timilsina, 2024)

Gambar 5 Layer-layer pada *detect block* YoloV8

Pada modul deteksi, terdapat 2 percabangan yang berfungsi pada dua hal, cabang pertama untuk melakukan prediksi *bounding box* dan cabang kedua untuk melakukan prediksi kelas objek. Hasil dari block ini akan memberikan *output* dengan ukuran 80x80x(5+C) yang mana C merupakan jumlah kelas objek yang perlu dideteksi.

1.2.7 Intersection Over Union

Intersection Over Union atau IOU merupakan metrics yang digunakan untuk mengevaluasi hasil dari permasalahan deteksi objek. Dalam tugas pendeteksian objek, area pada gambar yang telah dideteksi terdapatnya suatu objek ditandai dengan kotak yang disebut dengan *bounding box* sedangkan area pada gambar dimana objek terletak disebut dengan *ground truth box.* IOU merupakan perhitungan terhadap perbandingan lebar, tinggi, dan titik koordinat antara *bounding box* dan *ground truth box* seperti pada Gambar 6.



Sumber: (Rezatofighi dkk., 2019)

Gambar 6 Contoh Perbandingan dan Hasil IOU

Dapat dilihat pada Gambar 6, area yang memiliki garis berwarna hijau merupakan *ground truth box* dan area dengan garis berwarna hitam merupakan *bounding box.* Perbedaan antara *ground truth box* dan *bounding box* ini menentukan seberapa baik model yang dibangun dalam mendeteksi objek pada gambar. IOU menghitung perbedaan tersebut dengan menghitung area yang tumpang tindih antara kedua elemen tersebut sesuai persamaan (3) (Yu dkk., 2021):

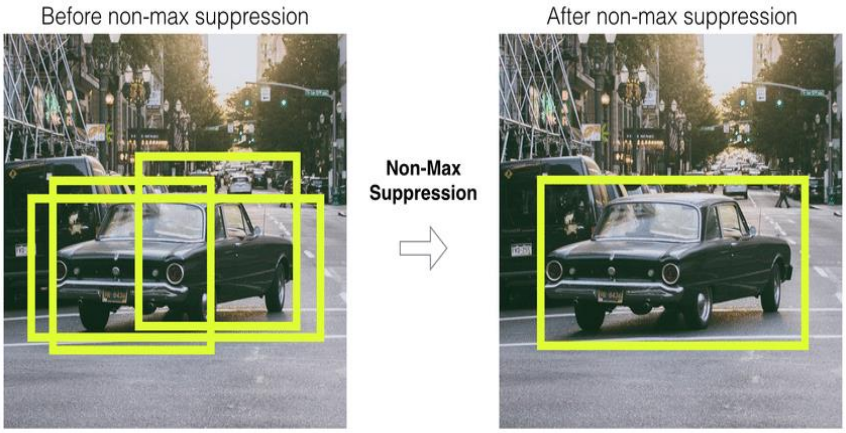
$$\text{IOU} = \frac{A \cap B}{A \cup B} \quad (3)$$

Keterangan:
A : *Ground Truth Box*
B : *Bounding Box*

Dari persamaan (3), IOU menghitung skor dengan membandingkan area titik potong antara kedua kotak dengan gabungan area dari kedua kotak tersebut. IOU mengeluarkan skor yang berada pada nilai 0 hingga 1. Nilai 0 mengartikan tidak ada area yang beririsan antara kedua *box* sedangkan nilai 1 mengartikan bahwa kedua *box* saling tumpeng tindih secara sempurna. Semakin besar skor yang dihasilkan oleh IOU maka semakin banyak area yang beririsan antara *ground truth box* dan *bounding box*. Maka hal itu mengindikasikan bahwa model yang telah dibangun sudah mampu mendeteksi keberadaan objek pada gambar dengan baik. Sebaliknya semakin kecil skor yang dihasilkan oleh IOU maka semakin sedikit area yang beririsan antara kedua *box* mengindikasikan bahwa model belum cukup mampu untuk melakukan pendeteksian objek pada gambar seperti yang diilustrasikan pada Gambar 6.

1.2.8 *Non Maximum Supression* (NMS)

Non Maximum Supression merupakan salah satu teknik yang digunakan setelah langkah proses deteksi untuk memperbaiki kesalahan deteksi dari model yang dikembangkan. Pada metode *deep learning*, khususnya yang menggunakan teknik *region proposal network* atau *sliding window*, model akan menjelajahi seluruh peta fitur dari suatu gambar, yang kemudian akan melakukan prediksi *bounding box* dengan skala dan aspek rasio yang berbeda. Teknik ini memungkinkan untuk membuat banyak *bounding box* dengan skor masing-masing pada suatu area. Permasalahan yang terjadi ketika terdapat beberapa *bounding box* yang memiliki skor yang cukup tinggi pada area yang berdekatan sehingga pada hasil akhirnya terdapat *false positive* pada deteksi akhir (Song dkk., 2019). Teknik NMS digunakan untuk mengatasi hal tersebut dengan memilih salah satu *bounding box* sebagai deteksi final seperti yang diilustrasikan pada



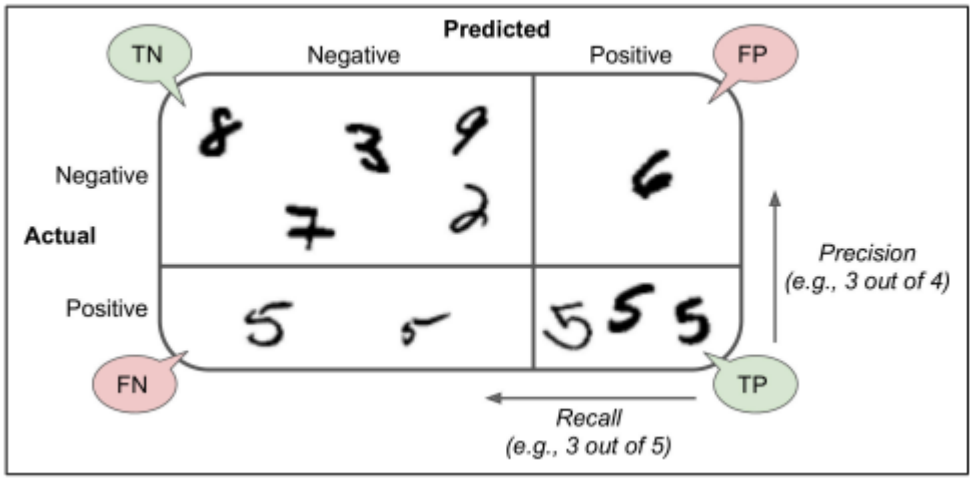
Sumber: (Jain & Nandy, 2019)

Gambar 7 Ilustrasi *Non-Max Suppression*

NMS bekerja dimulai dengan kumpulan box-box dengan masing-masing skor. NMS akan memilih box yang memiliki skor yang paling tinggi diantara yang lainnya. Setelah seleksi tersebut, NMS membandingkan dengan box yang lain dan akan menghilangkan box-box tersebut jika derajat tumpang tindih terhadap box dengan skor tertinggi melewati batas yang telah ditetapkan. Box-box yang tersisa dianggap sebagai deteksi objek akhir (Bodla dkk., 2017).

1.2.9 *Confusion Matrix*

Salah satu metrik yang digunakan untuk menilai seberapa baik model dalam menyelesaikan persoalan klasifikasi pada *computer vision* adalah dengan menggunakan *confusion matrix*. Prediksi-prediksi yang dikeluarkan oleh model digunakan untuk dibandingkan terhadap *ground truth*nya. *Confusion matrix* sendiri berupa tabel yang dikategorikan terhadap hasil prediksi dan kelas aktualnya (*ground truth*) pada masing-masing baris dan kolom. Biasanya, setiap kolom pada tabel merepresentasikan kelas aktual sedangkan baris pada tabel merepresentasikan kelas prediksi seperti yang ditunjukkan pada Gambar 8.



Sumber: (Geron, 2019)

Gambar 8 Tabel *Confusion Matrix* terhadap klasifikasi prediksi gambar angka 5

Dari ilustrasi Gambar 8, terdapat metriks lain yang bisa digunakan untuk mengevaluasi kinerja model yang disebut *precision*. *Precision* merupakan akurasi terhadap jumlah prediksi positif dari *output* model yang ditunjukkan pada persamaan (4).

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

Selain itu, metrik lain yang biasanya digunakan secara bersamaan dengan *precision* adalah *recall* yang juga biasa disebut sebagai *sensitivy* atau *true positive rate* (TPR). *Recall* merupakan rasio prediksi positif yang benar sesuai kelas aktual yang dideteksi oleh model. Persamaan dari *recall* dapat ditunjukkan pada persamaan (5).

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

Metriks yang lain disebut sebagai *accuracy* merupakan rasio jumlah keseluruhan hasil prediksi baik positif maupun negatif terhadap seluruh citra uji yang digunakan untuk menilai performa model. Persamaan dari *accuracy* ditunjukkan pada (6)

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{6}$$

1.2.10 Penelitian Terkait

Pertama. Penelitian yang dilakukan oleh Mohamad Darwiche pada tahun 2019 dapat melakukan pendeteksian terhadap *speed bump* menggunakan metode *profile lines* yang merupakan metode yang melihat pola periodik dari nilai piksel di suatu row pada suatu gambar. Pola yang dilihat adalah pola periodik piksel dengan

intensitas tinggi dan rendah yang terdapat pada *speed bump* yang di cat dengan pola *zebra cross*. Metode ini dapat mendeteksi *speed bump* dengan pola *zebra cross* dengan variasi jarak 5, 10, 15, dan 20 meter secara baik (Darwiche & El-Hajj-Chehade, 2020).

Kedua. Penelitian yang dilakukan oleh Netson Kennedy Babu C pada tahun 2020 yang mampu mendeteksi *speed bump* menggunakan algoritma Otsu dan operasi morfologikal pada citra digital secara terpisah. Algoritma Otsu dan operasi morfologikal memberikan rata-rata rasio deteksi *speed bump* sebesar 74.6% dan 85.8% sehingga membuat operasi morfologikal memiliki kinerja yang lebih baik dibanding Algoritma Otsu (Babu dkk., 2020).

Ketiga. Penelitian oleh Ervin Indra Nugrah pada tahun 2021 yang menggunakan Analisis Blob dan *Convolutional Neural Network* dalam pendeteksian polisi tidur. Polisi tidur yang dideteksi merupakan polisi tidur yang memiliki pola visual. Analisis blob mampu melakukan pendeteksian yang cukup baik pada polisi tidur yang memiliki pola visual yang cukup terang dan melakukan beberapa kesalahan deteksi pada polisi tidur dengan warna pudar dan tanpa pola visual. Perpaduan metode *Convolutional Neural Network* dengan Analisis Blob dapat mengatasi kelemahan Analisis blob dalam mendeteksi polisi tidur dengan warna pudar dan tanpa pola visual dengan tingkat akurasi sebesar 76% (Nugraha & Pranoto, 2021).

Keempat. Penelitian dengan judul "*Pedestrian Detection System using Yolov5 for Advanced Driver Assistance System (ADAS)"* yang dilakukan oleh Surya Michrandi Nasution menggunakan Yolov5 untuk mendeteksi pejalan kaki. Penggunaan Yolov5 dengan jumlah *epoch* sebesar 1000 memberikan *bounding box loss* sebesar 0.06. Sistem yang digunakan merupakan mini komputer Raspberry Pi sebagai unit proses utama hanya mampu memberikan kemampuan komputasi sebanyak 0.9 *frame per second* (Nasution & Dirgantara, 2023).

Kelima. Penelitian yang dilakukan oleh Lotfi Ezzedini dengan judul "*Analysis of the performance of Faster R-CNN and YOLOv8 in detecting fishing vessels and fishes in real time*" memberikan hasil bahwa algoritma YoloV8 memberikan hasil yang lebih baik dibandingkan dengan Faster-RCNN dalam mendeteksi jenis ikan dan kapal. YoloV8 mendapatkan nilai *precision* dan *recall* sebesar 76% dan 61% untuk deteksi ikan dan 87% dan 90% untuk deteksi kapal. Sedangkan Faster RCNN hanya mendapatkan nilai *precision* dan *recall* sebesar 68% dan 58% untuk deteksi ikan dan 76% dan 73% untuk deteksi kapal (Ezzeddini dkk., 2024).

1.3 Rumusan Masalah

Berdasarkan latar belakang diatas, dirumuskan beberapa masalah yang dihadapi pada penelitian ini:

1. Bagaimana cara atau langkah dalam melakukan pendeteksian *speed bump* dan *rumble strip* dengan menggunakan algoritma YoloV8?
2. Bagaimana performa model dengan algoritma YoloV8 dalam mendeteksi *speed bump* dan *rumble strip* pada variasi kondisi kecepatan kendaraan?

1.4 Tujuan Penelitian

Tujuan dari penelitian ini diantaranya:

1. Merancang dan membangun suatu sistem yang mampu melakukan pendeteksian terhadap *speed bump* dan *rumble strip* menggunakan algoritma YoloV8.
2. Mengetahui kinerja dari algoritma YoloV8 dalam mendeteksi *speed bump* dan *rumble strip*.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat sebagai berikut:

1. Menambahkan salah satu referensi terkait fitur dalam membangun sistem *Advance Driver Assistance System* (ADAS).
2. Sebagai bahan referensi ilmiah untuk penelitian selanjutnya di bidang terkait

1.6 Ruang Lingkup

Penelitian ini memiliki beberapa batasan, yaitu:

1. Pengambilan data dilakukan di siang hari saat cuaca cerah.
2. Data diambil menggunakan webcam dengan resolusi 2560 x 1440 dengan *frame rate* 30fps.
3. Data diambil dengan mengendarai mobil tanpa pengaruh kaca film riben.

BAB II

METODE PENELITIAN

2.1 Kerangka Berpikir

Masalah
Sistem ADAS memerlukan fitur-fitur yang mampu mengenali mayoritas objek yang hadir pada lingkungan jalanan. Kedua objek yang biasanya hadir pada jalan di Indonesia diantaranya adalah *speed bump* dan *rumble strip*. Kedua objek tersebut dipasang melintang sepanjang jalan dengan meninggikan sebagian badan jalan sehingga kendaraan perlu melewati objek tersebut dengan menurunkan kecepatan. Namun kedua objek tersebut khususnya *speed bump* dipasang tanpa adanya pola visual yang dapat dikenali sebagai *speed bump* sehingga terdapat resiko keamanan bila objek tersebut tidak diketahui keberadaannya dan dilalui dengan kecepatan tinggi ataupun menurunkan kecepatan kendaraan secara mendadak.

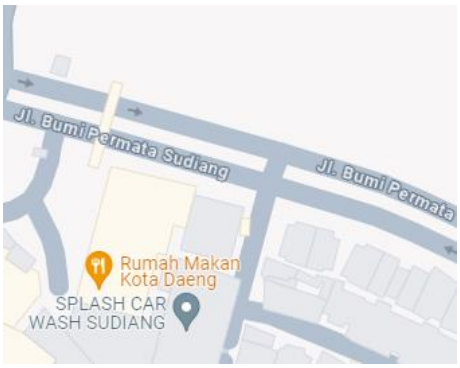
Tujuan
1. Merancang dan membangun suatu sistem yang mampu melakukan pengenalan objek *speed bump* dan *rumble strip*. 2. Mengetahui kinerja dalam tugas pendeteksian *speed bump* dan *rumble strip*.

Analisis
1. Menganalisis performa sistem dalam melakukan pendeteksian objek speed bump dan rumble strip pada jarak tertentu. 2. Menganalisis performa sistem dalam melakukan pendeteksian objek speed bump dan rumble strip pada kendaraan dengan kecepatan tertentu.

Hasil
Mendapatkan sistem yang mampu melakukan pendeteksian objek *speed bump* dan *rumble strip* secara optimal

2.2 Lokasi Penelitian

Penelitian ini dilaksanakan selama 11 bulan dimulai dari September 2023 hingga Juli 2024. Data-data yang digunakan selama penelitian diambil pada beberapa lokasi diantaranya di Kecamatan Biringkanaya tepatnya di Jl. Bumi Permata Sudiang, Jl. Perintis Kemerdekaan, Jl. Poros Bandara Baru Sultan Hasanuddin, Jl. Bahagia, Jl. KH. Abd. Jabbar Ashiry dan di Kecamatan Tamalanrea tepatnya di Jl. Jalur Lingkaran Barat. Tahapan pelaksanaan penelitian dilakukan di Laboratorium Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

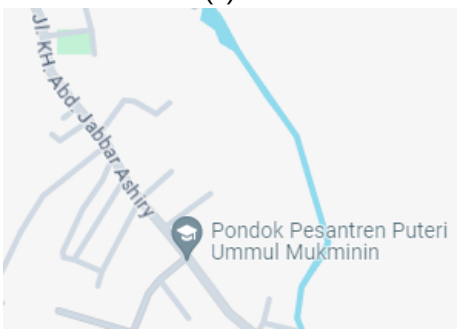


(a)

(b)

(c)

(d)



(e)

(f)

Gambar 9 Lokasi (a) Jl. Bumi Permata Sudiang, (b) Jl. Perintis Kemerdekaan, (c) Jl. Poros Bandara Baru, (d) Jl. Bahagia, (e) Jl. KH. Abd. Jabbar Ashiry, dan (f) Jl. Jalur Lingkaran Barat

2.3 Instrumen Penelitian

Terdapat 2 jenis komponen instrumen penelitian yang digunakan selama penelitian berlangsung yaitu perangkat lunak dan perangkat keras. Beberapa perangkat lunak dan perangkat keras yang digunakan selama penelitian ini diantaranya ialah:

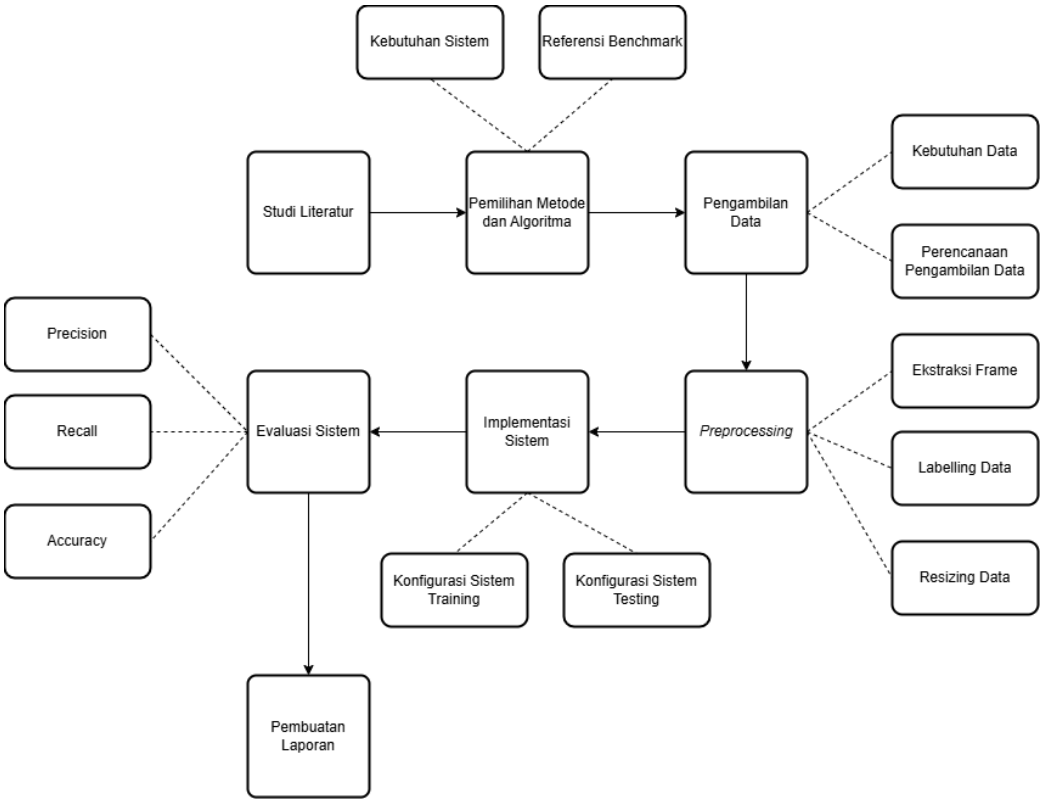
Tabel 1 Spesifikasi perangkat lunak

Komponen	Spesifikasi
OS	Windows 10
Bahasa	Python
Tools	Visual Studio Code, Kaggle Notebook, Microsoft Office, Roboflow, Google Colaboratory, OpenCV, PyGame

Tabel 2 Spesifikasi perangkat keras

Komponen	Spesifikasi
Laptop	Asus X441U, MSI GF63 Thin
Processor	IntelCore i3, IntelCore i7
RAM	8GB, 16GB
Webcam	NYK Nemesis A96

2.4 Tahapan Penelitian



Gambar 10 Tahap Penelitian

2.4.1 Studi Literatur

Tahap awal penelitian dilakukan dengan studi literatur dengan tujuan untuk mengumpulkan informasi-informasi terkini dari jurnal, artikel, dan penelitian-penelitian sebelumnya terkait terhadap permasalahan deteksi objek *speed bump* ataupun *rumble strip*.

2.4.2 Pemilihan Metode dan Algoritma

Sebelum menentukan metode atau algoritma yang digunakan, terdapat kebutuhan yang menjadi pertimbangan dalam membangun sistem.

Kebutuhan Sistem. Sistem perlu melakukan proses deteksi yang cepat sehingga mampu diimplementasikan pada lingkungan *realtime*. Jumlah *frame rate* yang ingin dicapai ketika melakukan pengujian secara realtime sebesar 8-9 fps. Selain itu, sistem juga perlu memiliki performa yang cukup baik dalam melakukan tugas pendeteksian objek *speed bump* dan *rumble strip* dengan meminimalisir kesalahan deteksi.

Referensi Benchmark. Terdapat 2 opsi metode *deep learning* untuk tugas pendeteksian objek diantaranya adalah YoloV8 dan Faster R-CNN. Namun berdasarkan dari beberapa penelitian seperti pada penelitian yang dilakukan oleh Akilesh Sharma dengan judul "*Comparative Performance of Yolov8, Yolov9,*

Yolov10 and Faster R-Cnn Models for Detection of Multiple Weed Species", didapatkan hasil bahwa waktu inferensi yang diperoleh YoloV8 dalam melakukan deteksi lebih cepat dibandingkan Faster R-CNN dengan waktu inferensi YoloV8 sebesar 23 ms per gambar sedangkan waktu inferensi Faster R-CNN sebesar 63,8 ms per gambar. Selain itu, pada penelitian yang dilakukan oleh Lotfi Ezzeddini dengan judul "*Analysis of the performance of Faster R-CNN and YOLOv8 in detecting fishing vessels and fishes in real time*" didapatkan hasil bahwa nilai *precision* dan *recall* untuk deteksi ikan dengan model Faster RCNN sebesar 68% dan 58% lebih kecil dibandingkan nilai *precision* dan *recall* untuk deteksi ikan dengan model YoloV8 yang sebesar 76% dan 61%. Sehingga dari referensi tersebut, penelitian ini mengimplementasikan algoritma YoloV8 untuk melakukan deteksi *speed bump* dan *rumble strip*. Selain itu, YoloV8 mendapatkan beberapa peningkatan performa dibandingkan versi dahulunya yaitu YoloV5, perubahan pada dari YoloV5 ke YoloV8 terdapat pada *CSPLayer*. Pada YoloV5, hanya output dari *bottleneck layer* yang terakhir digunakan, sedangkan pada YoloV8 yang disebut *C2F Layer*, seluruh *bottleneck layer* akan digunakan dan digabungkan dengan *layer convolution* sebelumnya. Penggabungan tersebut bertujuan untuk mengkombinasikan lebih banyak fitur-fitur pada *input image.* Penggunaan kembali fitur-fitur tersebut membuat model dapat memperkaya pola-pola atau berbagai informasi dari suatu objek sehingga dapat meningkatkan kemampuan model dalam konteks deteksi objek.

2.4.3 Pengambilan Data

Data pada penelitian ini menggunakan data primer yang diperoleh secara langsung dari beberapa lokasi penelitian seperti yang ditunjukkan pada Gambar 9. Terdapat beberapa perencanaan dan kebutuhan data diantaranya:

Kebutuhan Data. Data yang dibutuhkan merupakan data speed bump yang terbuat dari semen dan melintang sepanjang jalan secara utuh. Selain itu, data rumble strip yang dipasang berulang sebanyak lebih dari 3 kali juga diperlukan untuk menjaga konsistensi data yang dikumpulkan.

Perencanaan Pengambilan Data. Data dikumpulkan dalam rentang waktu bulan Agustus – Oktober tahun 2023. Data diambil dalam kondisi siang hari dengan cuaca cerah. Pengambilan data pada kondisi siang hari dan cuaca cerah dipertimbangkan agar cahaya yang masuk saat *webcam* menangkap citra atau gambar cukup sehingga objek *speed bump* dan *rumble strip* dapat terlihat jelas pada gambar.

2.4.4 *Preprocessing*

Data-data yang telah dikumpulkan akan dilakukan proses penyesuaian data agar model yang dibangun menerima data-data yang relevan dan sesuai dengan masukan yang diterima oleh model. Proses penyesuaian data yang dilakukan pada penelitian ini diantaranya:

Ekstraksi *frame*. Video dari proses pengambilan data akan dipecah menjadi *frame-frame* gambar. Video yang diambil memiliki frame rate sebesar 30 *frame per second*. Video tersebut dipecah dengan mengambil 5 *frame* dari setiap detik

video. Dari kumpulan *frame* tersebut diseleksi untuk mendapatkan *frame-frame* yang mengandung informasi terkait objek *speed bump* dan *rumble strip*.

***Labelling* data.** *Frame-frame* yang telah diekstrak akan dikumpulkan untuk memulai proses *labelling data*. Pelabelan *frame* dilakukan dengan menggunakan aplikasi web yaitu Roboflow dengan mengidentifikasi wilayah atau area pada gambar yang terdapat objek dan memberikan kategori kelas terhadap objek yang hadir pada wilayah tersebut.

***Resizing* data.** Data-data yang telah melalui proses *labelling* akan memasuki tahapan *resizing* atau penyesuaian ukuran gambar agar sesuai dengan format masukan yang dapat diterima oleh model dan menghemat penggunaan sumber daya selama berjalannya proses pembelajaran model. Model YoloV8 secara umum menggunakan ukuran gambar 640x640 untuk masukan sehingga perlu dilakukan penyesuaian gambar yang mulanya berukuran 2560 x 1440 menjadi 640x640.

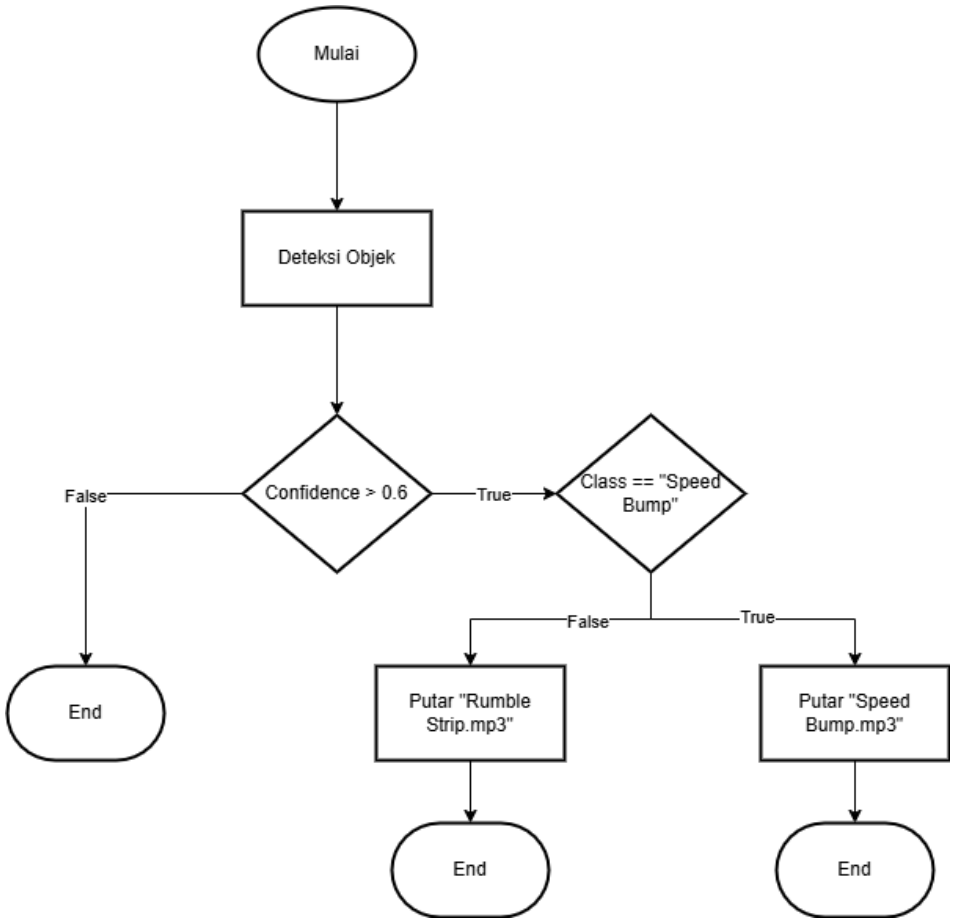
Penyesuaian ukuran gambar agar menghemat penggunaan sumber daya selama berjalannya proses pembelajaran model. Selain itu, karena data yang diambil dalam bentuk video, maka perlu dilakukan pemecahan video menjadi *frame-frame* citra gambar. Kemudian, dari *frame-frame* yang didapatkan, proses *labelling* atau anotasi citra perlu dilakukan untuk menentukan wilayah dari citra yang mengandung informasi objek dengan cara meletakkan *bounding box*.

2.4.5 Implementasi Sistem

Data yang melalui proses *preprocessing* akan dibagi menjadi 2 bagian data yaitu, data pelatihan dan data pengujian. Data pelatihan digunakan untuk melatih model yang pada penelitian ini mengimplementasikan algoritma YoloV8. Data pengujian digunakan untuk menilai performa model dalam memprediksi data yang tidak dikenali. Terdapat beberapa konfigurasi dalam proses pelatihan dan pengujian.

Konfigurasi sistem pada pelatihan. Pada saat melatih sistem, jumlah iterasi atau yang disebut *epoch* perlu dikonfigurasi untuk mendapatkan pelatihan model yang maksimal. Pada penelitian ini, jumlah *epoch* yang digunakan sebanyak 75, 100, 150, dan 200 *epoch*. Selain itu terdapat 3 fungsi *optimizer* yang digunakan pada penelitian ini yaitu fungsi *optimizer* Adam, AdamW dan SGD. Jumlah *batch_size* yang digunakan terdapat 2 yaitu sebesar 32 dan 64.

Konfigurasi sistem pada pengujian. Pada pengujian sistem, terdapat *CONFIDENCE_THRESH* yang merupakan batas minimum nilai *confidence* hasil deteksi yang dikeluarkan model. Pada penelitian ini, nilai *threshold* yang digunakan sebesar 0.6 sehingga hasil deteksi dengan nilai *confidence* dibawah nilai *threshold* akan diabaikan. Selain itu logika sistem untuk mengeluarkan output audio sesuai dengan kelas dari objek ketika terdapat deteksi yang dikeluarkan model ditunjukkan pada Gambar 11.



Gambar 11 Flowchart Output Audio

2.4.6 Evaluasi Sistem

Tahap ini dilakukan untuk menguji seberapa baik sistem dalam melakukan tugas pendeteksian pada objek *speed bump* ataupun *rumble strip*. Beberapa metrik digunakan untuk mengevaluasi sistem yang telah dibangun. Metriks yang digunakan diantaranya.

Precision. *Precision* merupakan rasio jumlah deteksi positif yang benar terhadap seluruh deteksi yang dikeluarkan oleh model. Persamaan dari *precision* ditunjukkan pada persamaan (4). *Precision* digunakan untuk mencari tahu seberapa besar persentase objek yang benar *speed bump* dan *rumble strip* dari keseluruhan objek yang dideteksi *speed bump* dan *rumble strip*.

Recall. *Recall* merupakan rasio jumlah deteksi positif yang benar terhadap seluruh data positif yang tersedia. Persamaan *recall* ditunjukkan pada persamaan (5). *Recall* digunakan untuk mencari tahu seberapa besar persentase objek yang diprediksi *speed bump* dan *rumble strip* dari keseluruhan data objek *speed bump* dan *rumble strip* yang sebenarnya.

***Accuracy*.** *Accuracy* merupakan rasio jumlah deteksi positif ataupun negatif yang benar terhadap seluruh data yang tersedia. Persamaan *accuracy* ditunjukkan pada persamaan (6). *Accuracy* digunakan untuk mencari tahu seberapa besar persentase

prediksi objek *speed bump* dan *rumble strip* dan bukan keduanya dari keseluruhan data yang di uji.

2.4.7 Pembuatan Laporan

Tahap terakhir pada penelitian ini yaitu melakukan penulisan laporan yang berisi hasil-hasil dari penelitian yang telah dilakukan yang dituangkan ke dalam bentuk tulisan skripsi.

2.5 Teknik Pengambilan Data

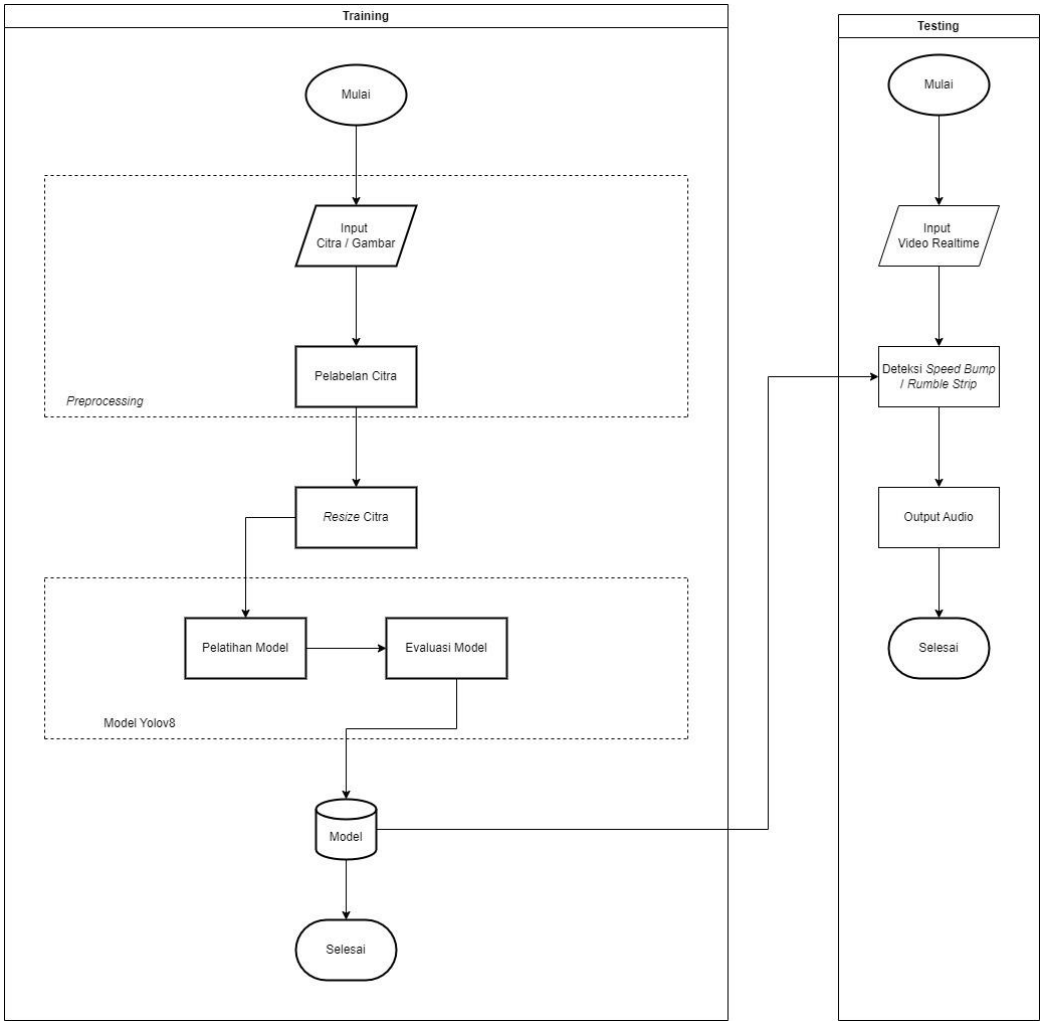
Data yang digunakan merupakan data yang diambil secara mandiri yang dikumpulkan dari beberapa titik lokasi yang berada di Kota Makassar, Sulawesi Selatan, sesuai yang ditunjukkan pada Gambar 9. Data dikumpulkan dengan menggunakan kamera *webcam* NYK Nemesis A96 dengan resolusi perekaman sebesar 2560 x 1440 pixel dan 30 *frame per second*. Data diambil dengan memasang kamera di belakang kaca spion dalam mobil seperti yang ditunjukkan pada Gambar 12.

Gambar 12 Penempatan kamera *webcam*

Pengambilan data di lakukan di siang hari pada pukul 7.00 hingga 16.00 dengan kondisi cuaca cerah. Kecepatan mobil ketika melakukan pengambilan data bervariasi antara 10 – 30 km/jam.

2.6 Perancangan Sistem

Sistem yang dirancang untuk mendeteksi *speed bump* atau *rumble strip* secara garis besar melalui 2 tahapan besar, yaitu *training* dan *testing* seperti yang ditunjukkan pada Gambar 13.



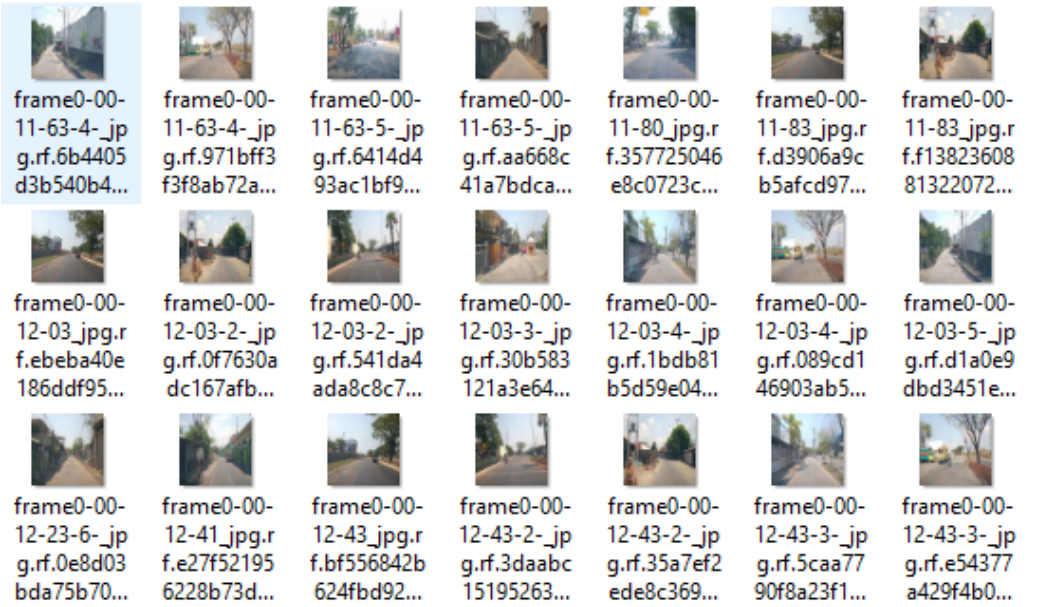
Gambar 13 Rancangan Sistem

Seperti yang diilustrasikan pada Gambar 13, terdapat beberapa tahapan pada tahap *training* diantaranya adalah input citra/gambar, pelabelan citra, *resize* citra, pelatihan model, dan evaluasi model.

2.6.1 Input Citra

Citra atau gambar yang menjadi masukan pada sistem merupakan citra yang berukuran 2560 x 1440 piksel. Citra ini didapatkan dari data yang diambil menggunakan video dengan resolusi yang sama dengan jumlah *frame* per detik yaitu 30 *frame per second*. Video ini kemudian diekstrak dengan mengambil 5 *frame* per detik menggunakan

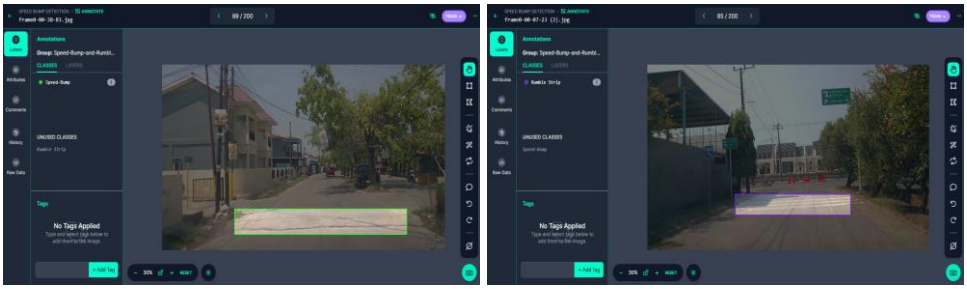
bantuan bahasa pemrograman python. Gambar 14 menunjukkan sampel data hasil dari ekstraksi video ke bentuk *frame*.



Gambar 14 Sampel Data Hasil Ekstraksi Video

2.6.2 Pelabelan Citra

Citra-citra yang telah diekstraksi dari video akan di lanjutkan ke dalam proses *labelling* atau anotasi citra. Citra di label berdasarkan tipe atau klasifikasi objeknya yang dalam penelitian ini terdapat 2 yaitu *speed bump* dan *rumble strip*. Proses anotasi dilakukan dengan bantuan perangkat lunak yaitu *roboflow*. Gambar 15 menunjukkan beberapa contoh proses pelabelan untuk masing-masing kelas objek.



Gambar 15 Pelabelan Citra menggunakan Roboflow

2.6.3 Resize Citra

Citra yang telah dilabeli akan melalui proses pengaturan dimensi. Citra yang awalnya berukuran 2560 x 1440 piksel akan diubah ukurannya menjadi 640 x 640 piksel untuk YOLOv8 dengan alasan untuk mengurangi beban komputasi ketika melakukan proses pembelajaran model namun dengan tetap mempertimbangkan untuk tidak

menghilangkan sebanyak mungkin fitur-fitur yang hadir pada objek. Gambar 16 merupakan potongan kode untuk melakukan proses *resize* citra.

```
IMAGES_PATH = dataset.location + "/train/images"
RESIZE_IMG_WIDTH = 640
RESIZE_IMG_HEIGHT = 640

for file in os.listdir(IMAGES_PATH):
    img = cv2.imread(file)
    img = cv2.resize(img, (RESIZE_IMG_WIDTH, RESIZE_IMG_HEIGH))
    cv2.imwrite(IMAGES_PATH + "/" + file, img)
```

Gambar 16 Potongan Kode Proses *Resize* Citra

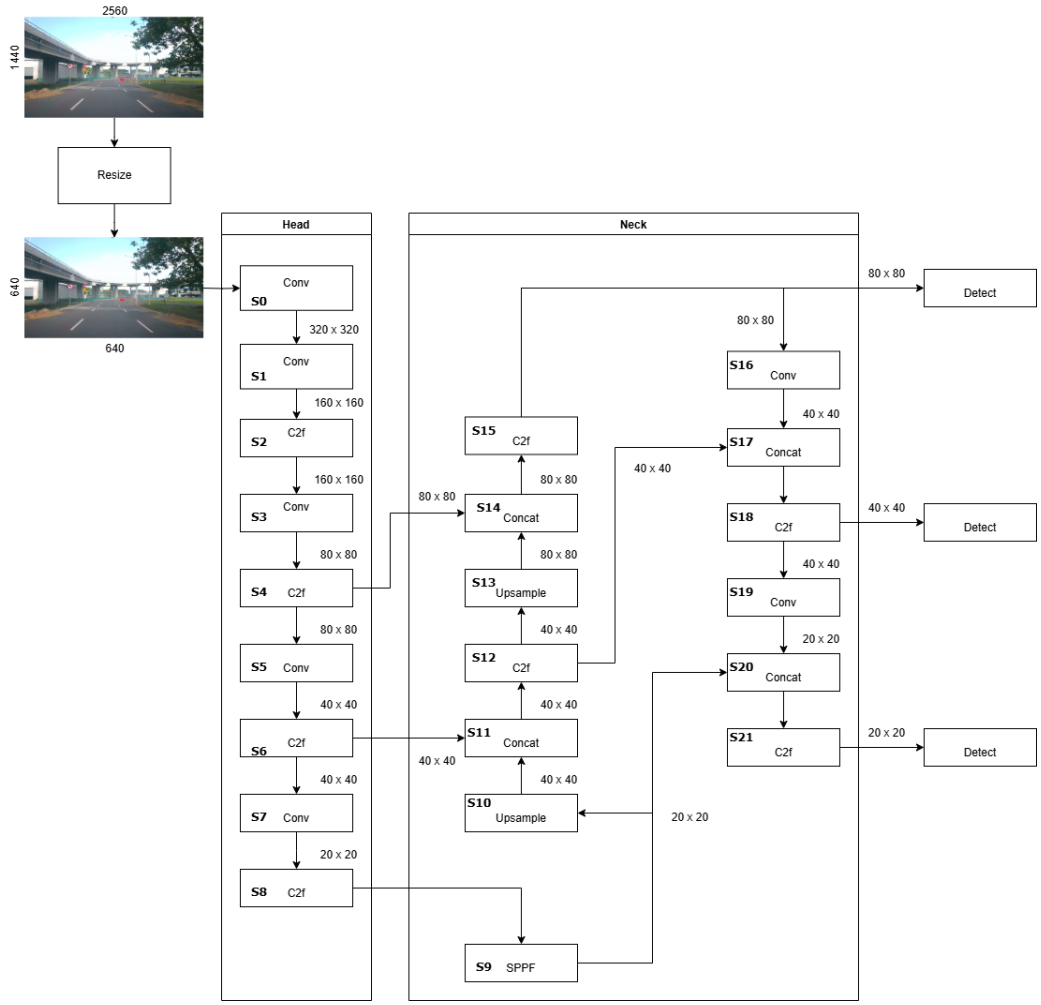
2.6.4 Pelatihan Model

Setelah data disiapkan dengan melalui proses pelabelan dan resize, data kemudian akan melalui proses pelatihan untuk model. Pada penelitian ini, model arsitektur yang digunakan adalah YOLOv8. Pada YoloV8 terdapat beberapa parameter yang digunakan selama proses penelitian. Parameter yang digunakan dapat dilihat pada Tabel 3.

Tabel 3 Parameter Pelatihan YoloV8

Parameter	Argumen
Model	YoloV8s.pt (Small)
Epochs	75, 100, 150, 200
Optimizer	Adam, AdamW, SGD
Batch Size	32, 64
Learning Rate	10^{-3}
Image Size	640

Gambar 17 menunjukkan ilustrasi diagram dari arsitektur YoloV8.

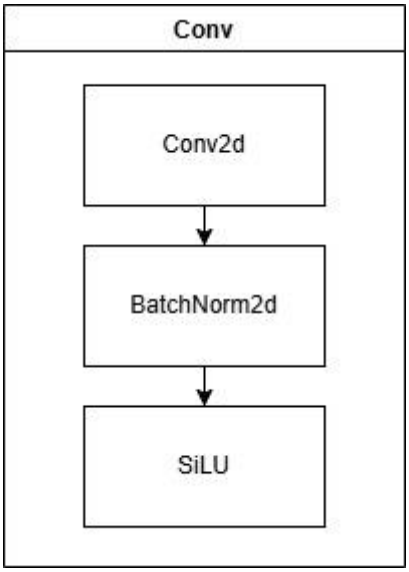


Gambar 17 Diagram Arsitektur YoloV8

Citra pada data *training* yang telah melalui proses *resizing* dengan ukuran sebesar 640x640 menjadi input pada YoloV8. Citra tersebut akan melalui tahapan *head* untuk mendapatkan peta fitur. Peta fitur merupakan representasi visual dari informasi-informasi karakteristik atau ciri objek yang terdapat pada gambar. Pada tahapan head, citra inputan akan melalui serangkaian proses konvolusi untuk mengubahnya menjadi peta fitur dan akan mengubah dimensi dari peta fitur hingga berukuran 20x20. Peta-peta fitur tersebut akan melalui tahapan selanjutnya yaitu tahapan *neck*.

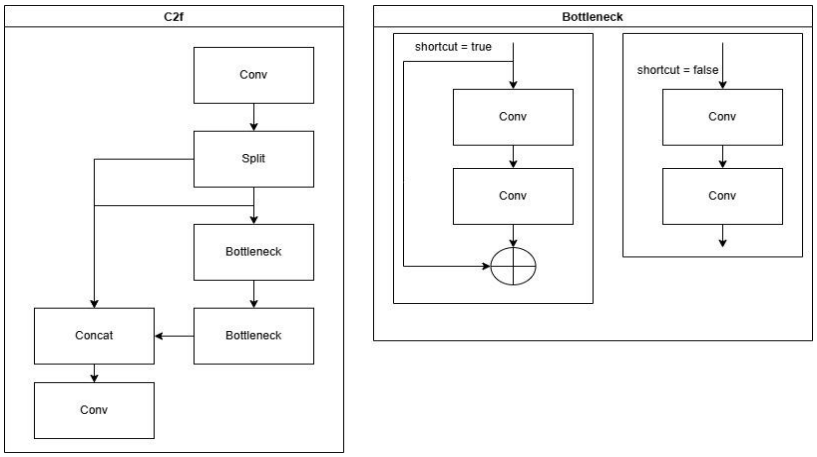
Pada tahapan *neck*, Peta fitur dengan ukuran 20x20 dari tahapan *head* akan melalui proses *upsampling* yang akan mengubah dimensi peta fitur menjadi 40x40 dan akan melalui proses *concat* untuk digabungkan dengan peta fitur berukuran 40x40 dari tahapan *head*. Proses *upsampling* dan *concat* terus berlangsung hingga peta fitur menjadi berukuran 80x80. Masing-masing peta fitur berukuran 80x80 , 40x40, dan 20x20 pada tahapan *neck* akan melalui proses deteksi untuk menghasilkan deteksi kelas dan lokasi dari objek yang hadir pada citra inputan. Adapun beberapa serangkaian blok yang dilewati input gambar menjadi peta fitur berukuran 80x80, 40x40 dan 20x20 diantaranya.

Conv. Blok Conv berisikan blok Conv2d, BatchNormalization, dan SiLU seperti yang ditunjukkan pada
Gambar 18 Block Conv. Conv2d melakukan proses konvolusi sedangkan block BatchNorm2d digunakan untuk meningkatkan stabilitas saat training dengan melakukan normalisasi sedangkan blok SiLU merupakan *activation function* yang menggunakan *Sigmoid Linear Unit*.



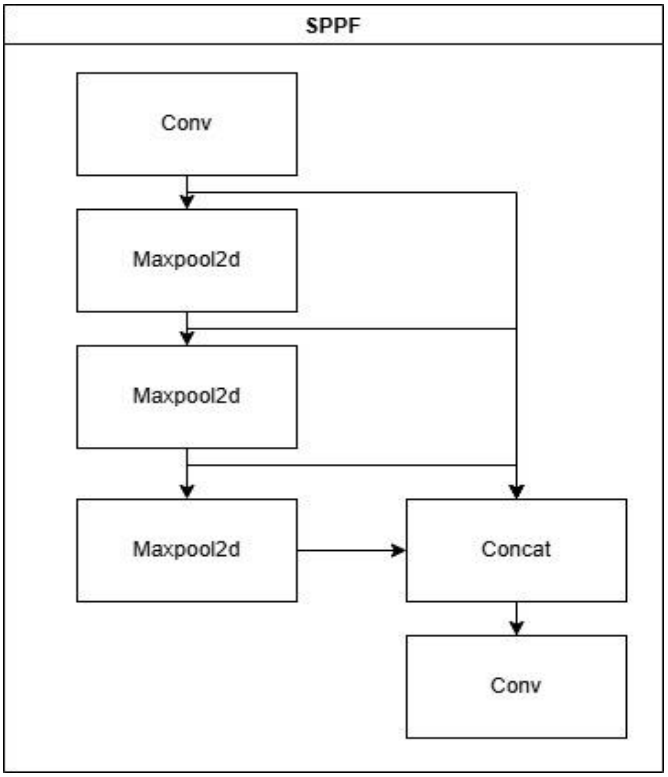
Gambar 18 Block Conv

C2f. Blok C2f berisikan blok Conv dan *bottleneck* seperti yang ditunjukkan pada Gambar 19. Pada blok C2f, hasil dari blok Conv akan dibagi sebagiannya menuju ke dalam blok *bottleneck* dan sebagian lainnya menuj =u ke blok *concat* untuk digabungkan dengan hasil dari blok *bottleneck*. Pada blok *bottleneck*, terdapat parameter *shortcut* untuk mengindikasikan input yang diterima melewati 2 blok conv pada blok *bottleneck*.



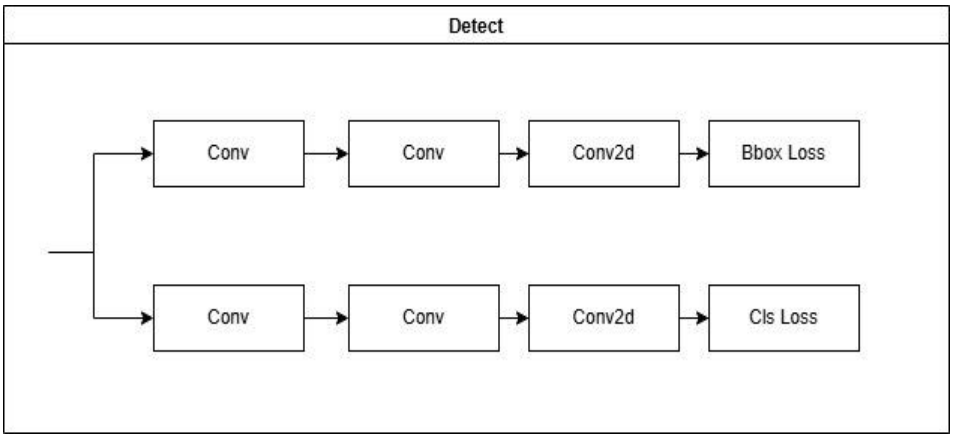
Gambar 19 Block C2f

SPPF. Blok SPPF berisikan 2 blok conv dan 3 blok maxpool2d yang ditunjukkan pada
Gambar 20. Hasil dari blok Conv dan setiap blok Maxpool2 akan memasuki blok Concat untuk digabungkan untuk melewati blok conv.



Gambar 20 Blok SPPF

***Detect*.** Blok *detect* terdiri dari 2 blok conv dan 1 conv2d seperti yang ditunjukkan pada Gambar 21. Blok *detect* bercabang menjadi dua dengan cabang pertama bertugas untuk melakukan prediksi terhadap *bounding box* objek dan cabang kedua bertugas untuk melakukan klasifikasi terhadap objek. Hasil prediksi *bounding box* dan klasifikasi kelas dihasilkan dari blok conv2d dari masing-masing cabang.



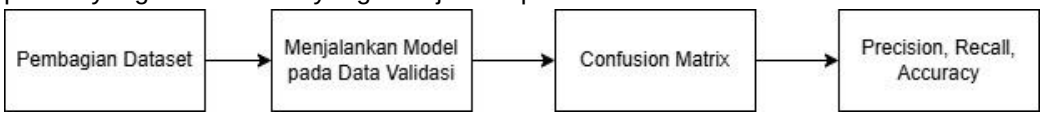
Gambar 21 Blok *Detect*

***Upsample*.** Blok *Upsample* digunakan untuk menaikkan resolusi peta fitur sebelum melakukan proses *concat*. Peta fitur yang akan melalui proses *concat* harus memiliki ukuran yang sama sehingga perlu dilakukan proses penyamaan ukuran melalui blok *Upsample*.

***Concat*.** Blok *Concat* merupakan blok yang melakukan penggabungan dua peta fitur tanpa merubah resolusi dari peta fitur tersebut. Penggabungan ini dilakukan dengan tujuan untuk memperoleh informasi atau fitur-fitur yang lebih beragam untuk memudahkan proses deteksi.

2.6.5 Evaluasi Model

Model yang telah dibangun kemudian dievaluasi dengan dua skenario pengujian yaitu pengujian dengan gambar statis dan pengujian secara *realtime*. Kedua pengujian melalui proses yang sama sesuai yang ditunjukkan pada Gambar 22



Gambar 22 Proses Evaluasi Model

Pembagian Dataset. Dataset dibagi menjadi 3 bagian yaitu, *training*, validasi, dan *testing*. Pada penelitian ini, jumlah data yang disiapkan untuk masing-masing data validasi dan data *testing* secara berturut-turut sebanyak 197 citra dan 249 citra.

Menjalankan Model. Data validasi yang telah dipisah dari data *training* kemudian digunakan untuk dijalankan sebagai bahan uji dalam melihat bagaimana performa model dari hasil latih yang telah dilakukan.

***Confusion Matrix*.** Hasil deteksi yang dikeluarkan oleh model terhadap data validasi akan dibandingkan hasilnya terhadap label yang sebenarnya dengan menggunakan *confusion matrix* untuk mendapatkan keterangan seperti *True Positive*, *True Negative*, *False Positive*, dan *False Negative* seperti yang ditunjukkan pada Gambar 8.

Nilai Metriks. Setelah membagi hasil prediksi pada data validasi dengan menggunakan *confusion matrix*, hasil prediksi akan dihitung untuk masing-masing

nilai *True Positive*, *True Negative*, *False Positive*, dan *False Negative* untuk mendapatkan nilai *precision*, *recall*, dan *accuracy* yang bisa didapatkan dengan menggunakan secara berturut-turut persamaan (4), (5), dan (6).

Selain itu, setelah melakukan pelatihan model, model juga perlu dievaluasi untuk mencegah terjadinya *overfitting*. Analisis terhadap nilai *loss* yang diantaranya adalah *training loss* dan *validation loss* perlu dilakukan untuk menilai trend penurunan kedua nilai *loss* dalam mencari tahu indikasi-indikasi yang menyatakan terjadinya *overfitting* selama proses *training* berjalan.

Perbedaan antara pengujian gambar statis dan pengujian secara realtime, pada pengujian dengan gambar statis, gambar yang digunakan untuk diprediksi merupakan gambar validasi yang tidak masuk dalam proses pelatihan model yang kemudian dari hasil prediksi tersebut akan dihitung nilai *precision* dan *recall*, sedangkan pada pengujian secara *realtime*, label hasil prediksi diambil dari video hasil rekam pengujian secara *realtime* yang kemudian juga akan dihitung nilai *precision* dan *recall*. Selain itu, dilakukan juga pengujian performa model pada beberapa kecepatan kendaraan. Kecepatan kendaraan yang diuji pada penelitian ini yaitu 20 km/jam, 30km/jam, dan 40km/jam. Jarak terjauh dari deteksi pertama kali yang dikeluarkan model juga dipertimbangkan pada pengujian secara *realtime*.