BAB 3
METODE PENELITIAN

Pada Bab 3 mengulas tentang metodologi penelitian, akan dibahas beberapa aspek termasuk perangkat yang digunakan untuk penelitian serta urutan penelitian yang meliputi flowchart dan diagram blok yang menjelaskan rincian penelitian ini.

3.1 ALAT YANG DIGUNAKAN

Studi ini memanfaatkan perangkat lunak untuk merancang dan menganalisis hasil keakuratan, penelitian kali ini merancang sebuah sistem simulasi pengolahan citra untuk sterilisasi pada jalur Transjakarta dengan menggunakan Google Collaboratory. Berikut adalah rincian mengenai karakteristik perangkat yang digunakan dalam pengembangan sistem ini, yaitu:

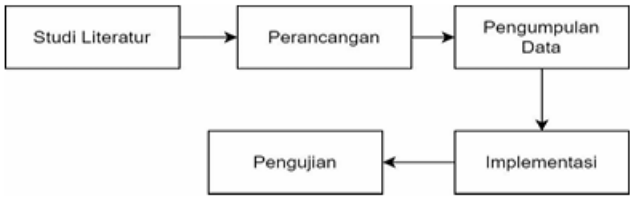
Tabel 3.1 Alat Yang Digunakan

Hardware	***Software***
Laptop HP 15s-Fq2642TU (*Processor intel core* i5, RAM 8 GB)	Sistem Operasi *Windows* 11 *Home Single Language* 64-bit
-	*Google Collaboratory*
-	*Roboflow*

Tabel 3.1 menjelaskan perangkat keras dan lunak yang digunakan pada penelitian ini. Untuk perangkat keras yang digunakan merupakan *Laptop* HP 15s-Fq2642TU (*Processor intel core* i5, RAM 8 GB) dan perangkat lunak yang digunakan terdiri dari Sistem Operasi *Windows* 11 *Home Single Language* 64-bit, *Google Collaboratory, Roboflow*.

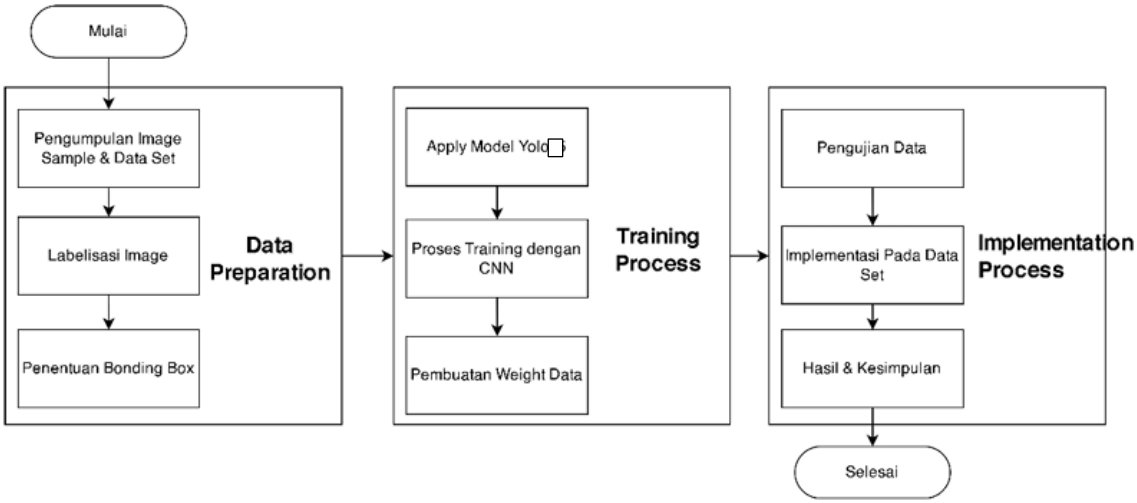
3.2 TAHAPAN PENELITIAN

Tahap penelitian digunakan Untuk memahami dan menganalisis isu penelitian, data yang relevan dengan topik penelitian dikumpulkan dan dianalisis sesuai dengan langkah-langkah yang telah ditetapkan. Penulis mengikuti alur kerja yang sudah ditetapkan guna mendapatkan hasil yang diinginkan. Berikut adalah diagram yang menggambarkan urutan langkah-langkah penelitian yang dilakukan dalam pembuatan sistem deteksi dan klasifikasi kendaraan untuk sterilisasi jalur TransJakarta menggunakan algoritma YOLO yang dijelaskan pada gambar 3.1 berikut.



Gambar 3.1 Alur Penelitian

Pada gambar 3.1 menjelaskan alur penelitian ini yang dimulai dari studi literatur untuk mencari referensi yang berkaitan dengan penelitian ini. Kemudian tahapan selanjutnya adalah perancangan sistem. Kemudian pengumpulan data atau persiapan data dilakukan dengan mengambil sampel data dan dataset yang akan digunakan, yang terdiri dari gambar atau foto terkait untuk deteksi objek kendaraan. Sumber data didapatkan dari sumber data pribadi.



Gambar 3.2 Tahapan Penelitian

Pada gambar 3.2 menjelaskan mengenai tahapan penelitian pembuatan sistem deteksi objek kendaraan untuk sterilisasi jalur TransJakarta menggunakan algoritma YOLO. Setiap gambar akan dilabeli dan diberi kotak pembatas yang berisi label jenis atau kelas kendaraan. Proses pelatihan bertujuan untuk menghasilkan bobot data atau hasil prediksi melalui proses pelatihan. Tahap implementasi dilakukan dengan menguji dan menerapkan model pada lokasi terkait menggunakan gambar yang diambil melalui kamera ponsel dengan posisi atau sudut pandang tidak bergeser – geser serta melakukan analisis dan memberikan Kesimpulan dari hasil pengujian yang telah dilakukan.

3.2.1 Studi literatur

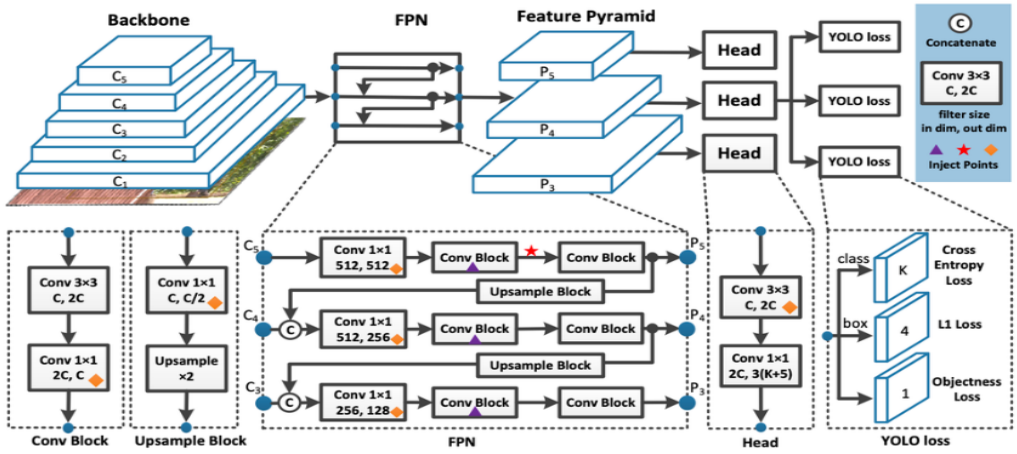
Melakukan tinjauan pustaka dengan cara mempelajari beberapa sumber-sumber yang relevan dengan penelitian ini digunakan sebagai basis teoritis untuk mendukung penelitian. Pengkajian literatur dilakukan dengan mengeksplorasi berbagai karya tulis yang berkaitan dengan deteksi dan klasifikasi kendaraan. Proses pencarian literatur dilakukan melalui *Google Scholar*, *Google Books*, dan sumber lainnya dengan menggunakan kata kunci yang relevan dengan fokus penelitian ini.

3.2.2 Metode Yang Diusulkan

Adapun pengembangan sistem yang dipertimbangkan untuk diterapkan dalam penelitian ini, metode penelitian yang dipilih adalah YOLO untuk mengembangkan sistem, karena metode ini mampu secara efisien mengidentifikasi dan menghitung kepadatan kendaraan dalam gambar. YOLO memanfaatkan jaringan saraf untuk melakukan deteksi objek dalam satu langkah, meningkatkan efisiensi proses dengan menganalisis pola warna RGB dari gambar yang telah disegmentasi. YOLO terkenal dengan kemampuannya yang superior dalam deteksi objek, pelacakan yang akurat, serta keamanan yang kuat. Dengan menggunakan library YOLO dalam proses deteksi objek kendaraan, dapat dipastikan identifikasi dilakukan secara cepat dan akurat.

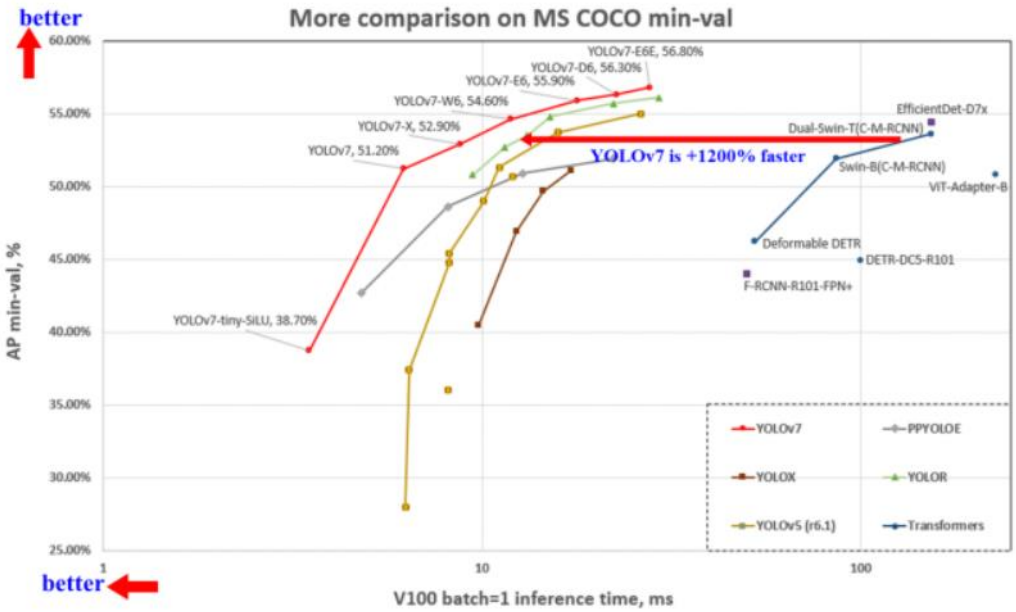
3.2.3 Arsitektur Sistem

Pada penelitian ini digunakan model arsitektur YOLOv7 bertujuan untuk dapat meningkatkan kemampuan YOLOv7 dalam deteksi objek kendaraan dalam gambar *grayscale*.



Gambar 3.3 Arsitektur YOLOv7 [27]

Pada gambar 3.3 menjelaskan bahwa YOLOv7 memberikan akurasi deteksi objek real-time yang jauh lebih baik tanpa meningkatkan biaya inferensi. Seperti yang ditunjukkan sebelumnya dalam tolok ukur, jika dibandingkan dengan detektor objek lain yang dikenal, YOLOv7 dapat secara efektif mengurangi sekitar 40% parameter dan 50% komputasi deteksi objek real-time terkini, dan mencapai kecepatan inferensi yang lebih cepat dan akurasi deteksi yang lebih tinggi. Secara umum, YOLOv7 menyediakan arsitektur jaringan yang cepat dan kuat yang menyediakan metode integrasi fitur yang lebih efektif, kinerja deteksi objek yang lebih akurat, fungsi kerugian yang lebih tangguh, dan peningkatan efisiensi proses penugasan label dan pelatihan model. Hasilnya, YOLOv7 membutuhkan perangkat keras komputasi yang jauh lebih murah daripada model pembelajaran mendalam lainnya. Model ini dapat dilatih jauh lebih cepat pada kumpulan data kecil tanpa bobot yang telah dilatih sebelumnya [27].



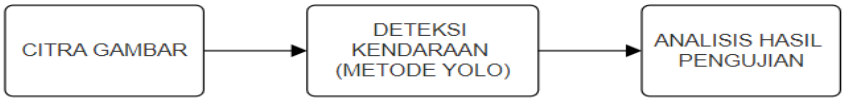
Gambar 3.4 Performa YOLO Dengan Versi Sebelumnya [27]

Gambar 3.4 menjelaskan bahwa dalam setiap rilis versi terbaru dari YOLO dimana pada gambar diatas YOLOv7 adalah versi tercepat yaitu 1200% lebih cepat dari beberapa versi YOLO lainnya. Dibandingkan dengan model *cascade – mask R-CNN* yang berperforma terbaik, YOLOv7 mencapai akurasi 2% lebih tinggi pada kecepatan inferensi yang meningkat secara dramatis (509% lebih cepat). Hal ini mengesankan karena versi R-CNN tersebut menggunakan

arsitektur multi – langkah yang sebelumnya mencapai akurasi deteksi yang jauh lebih tinggi dibandingkan arsitektur detector satu tahap. YOLOv7 mengungguli YOLOR, YOLOx, *scaled-YOLOv4*, YOLOv5, DETR, ViT, *Adapter-B*, dan banyak lagi algoritma deteksi objek lainnya dalam hal kecepatan dan akurasi. Dibandingkan dengan YOLOv4, YOLOv7 mengurangi jumlah parameter sebesar 75%, memerlukan komputasi 36% lebih sedikit, dan mencapai presisi rata-rata 1,5% lebih tinggi. Dibandingkan dengan versi YOLOv4-tiny yang dioptimalkan untuk edge, YOLOv7-tiny mengurangi jumlah parameter sebesar 39%, sekaligus mengurangi komputasi sebesar 49%, sekaligus mencapai presisi rata-rata yang sama. Dibandingkan dengan YOLOR, YOLOv7 mengurangi jumlah parameter sebesar 43%, memerlukan komputasi 15% lebih sedikit, dan mencapai presisi rata -rata 0,4% lebih tinggi. Saat membandingkan YOLOv7 dengan YOLOR menggunakan resolusi input 1280, YOLOv7 mencapai kecepatan inferensi 8 FPS lebih cepat dengan peningkatan tingkat deteksi (+1% AP).

3.2.4 Blok Diagram Sistem

Pada sistem yang dibangun, Input utama yang digunakan dalam aplikasi ini merupakan sebuah citra yang menampilkan kondisi jalur Transjakarta dengan berbagai jenis kendaraan. Citra ini akan diolah menggunakan algoritma YOLO untuk mendeteksi objek berdasarkan *dataset* yang telah dibuat dan dilatih sebelumnya akan dievaluasi hasil pelatihannya menggunakan algoritma untuk menilai sejauh mana objek dalam citra cocok dengan objek dalam *dataset* melalui prediksi geometris.

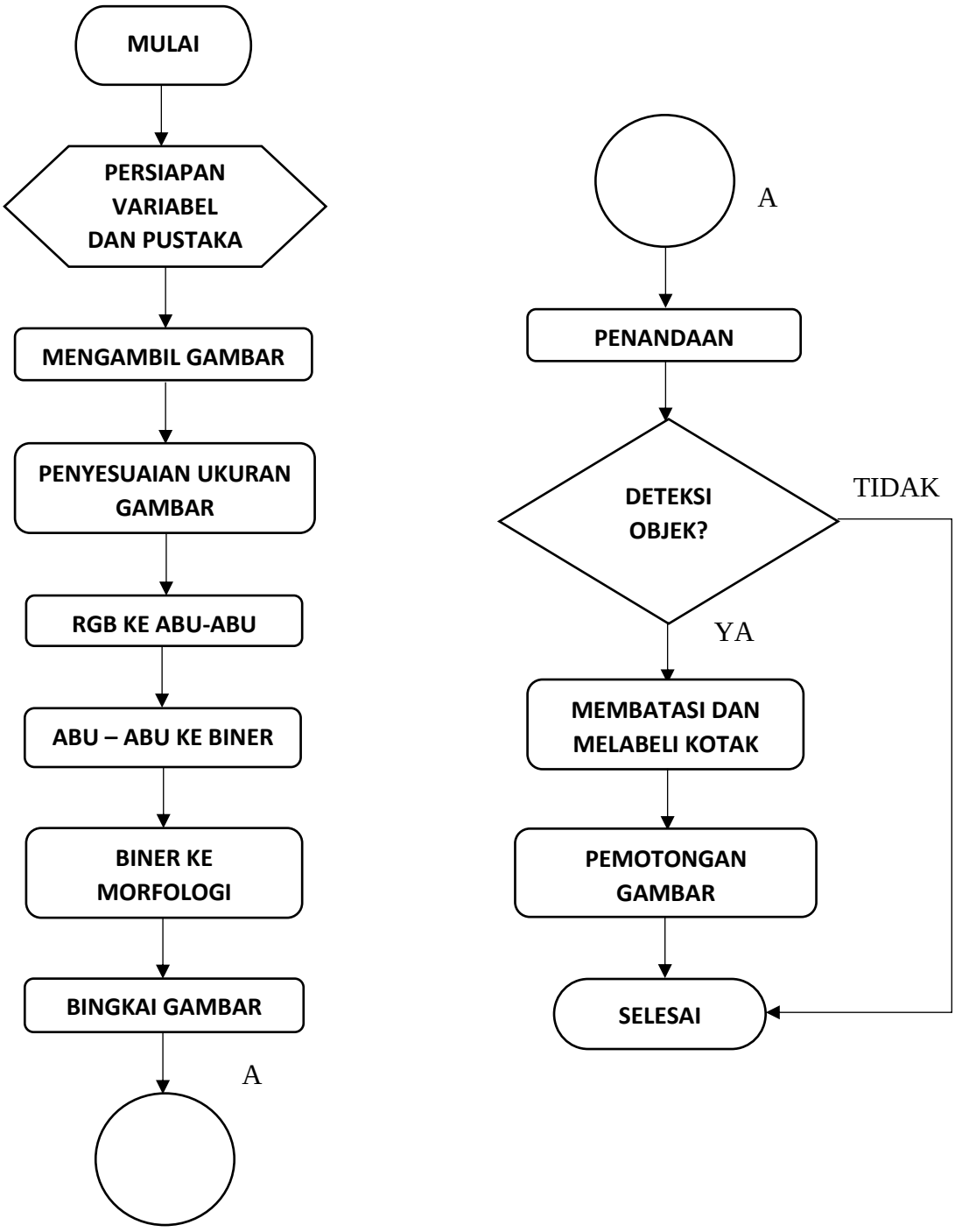


Gambar 3.5 Blok Diagram Sistem

Pada gambar 3.5 menjelaskan blok diagram sistem dari penelitian ini dimana citra gambar yang terdiri dari berbagai macam kendaraan akan dideteksi menggunakan YOLO. Ketika suatu objek memiliki probabilitas yang sesuai dengan *dataset*, YOLO akan menetapkan sebuah kotak pembatas pada objek tersebut dan memberikan label yang sesuai dengan objek yang terdeteksi. Setelah objek terdeteksi maka akan dilakukan analisis hasil pengujian.

3.2.5 *Flowchart* Sistem

Diagram alir sistem adalah representasi visual dari urutan operasi atau algoritma dalam sebuah program, yang memanfaatkan berbagai pustaka sistem yang tersedia. Detail *flowchart* ini dapat ditemukan pada gambar 3.2 yang terlampir di bawah ini.

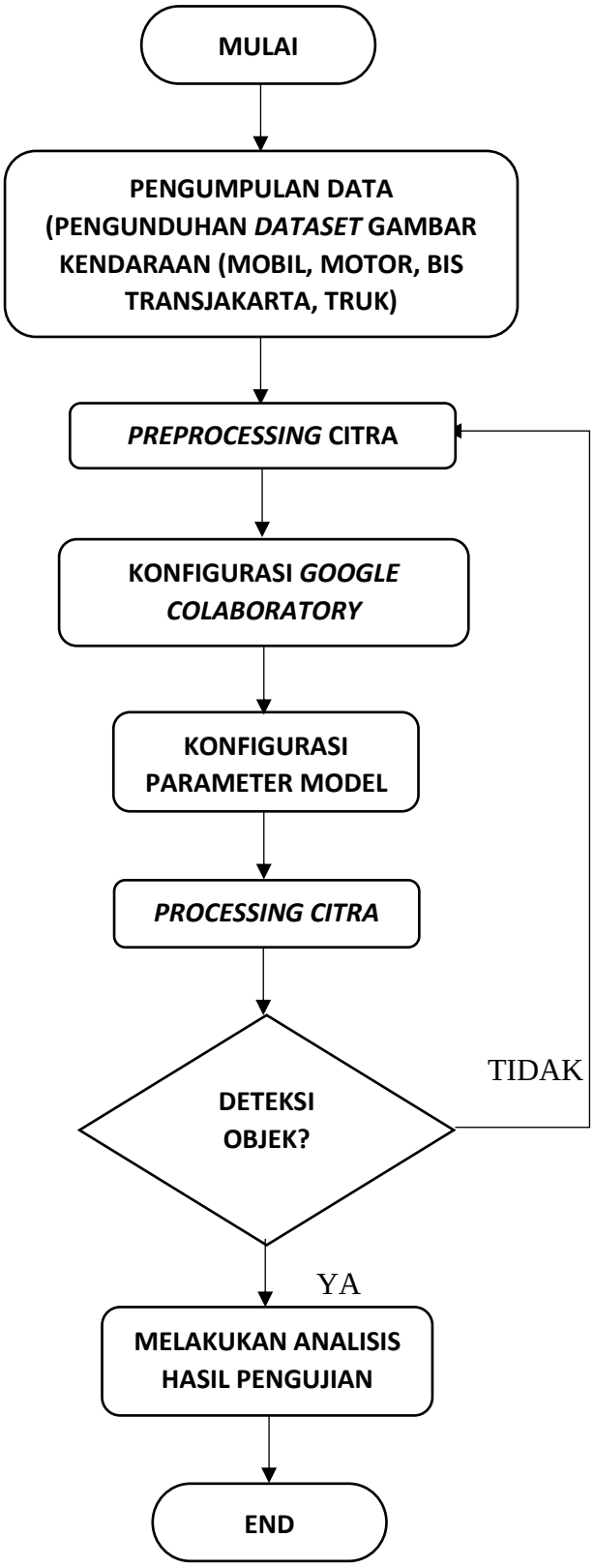


Gambar 3.6 *Flowchart* Sistem

Gambar 3.6 menunjukkan langkah-langkah persiapan variabel dan pustaka, diikuti dengan proses *Capture Frame* yang berfungsi untuk mengambil gambar. Setelah itu, gambar diubah ke dalam ruang warna RGB diubah ukurannya *(resize)* untuk mengurangi resolusi piksel. Hasil dari perubahan ukuran tersebut kemudian diubah menjadi citra *Greysca*le, yang selanjutnya diproses menjadi citra *binary*. Citra *binary* tersebut diperlakukan dengan proses morfologi untuk menghilangkan *noise*, dan kemudian dijalankan melalui proses grid untuk konvolusi. Keseluruhan proses ini dikenal sebagai *YOLO Detection*, di mana jika objek terdeteksi, proses selanjutnya melibatkan *bounding box* dan pemberian label. Namun, jika tidak terdeteksi, maka proses *Capture Frame* akan diulang.

3.2.6 Diagram Alir Deteksi Objek Dengan Algoritma YOLO

Pada Gambar 3.7 dibawah ini menunjukkan proses pegolahan citra menggunakan algoritma YOLO. Pada gambar 3.7 menjelaskan mengenai tahapan pengerjaan dalam bentuk diagram alir yang dimana pada tahapan pertama yang dilakukan adalah pengumpulan data berupa dataset dari kendaraan (Mobil, Motor, Bis TransJakarta) tahapan kedua yaitu melakukan preprocessing citra menggunakan algoritma YOLO, tahapan ketiga yaitu melakukan konfigurasi pada software *google colaboratory*, tahap keempat yaitu melakukan konfigurasi parameter model untuk menentukan tingkat akurasi, tahap kelima yaitu melakukan processing citra dengan algoritma YOLO, kemudian tahap keenam melakukan pengujian dan apabila hasil tidak sesuai maka Kembali ke tahap konfigurasi parameter model. *Preprocessing* dilakukan untuk melakukan penyesuaian ukuran citra untuk meningkatkan kualitas performa model YOLO. Kemudian konfigurasi pada *google colaboratory* sebagai *notebook* untuk menjalankan sistem deteksi objek kendaraan. Kemudian konfigurasi parameter model dilakukan pada API *Roboflow* yang diantaranya terdiri dari pengaturan *confidence score* atau nilai kepercayaan suatu sistem deteksi dalam mendeteksi objek kendaraan yang pada penelitian ini nilai *confidence score* yang akan tampil di sistem deteksi adalah yang memiliki nilai 50% keatas, jika nilai kurang dari 50% maka tidak akan ditampilkan dalam sistem. Kemudian melakukan *training dataset* yang berfungsi untuk melatih sistem untuk mempelajari gambar sehingga pada saat pengujian sistem dapat bekerja dengan baik



Gambar 3.7 Diagram Alir Deteksi Objek Dengan Algoritma YOLO

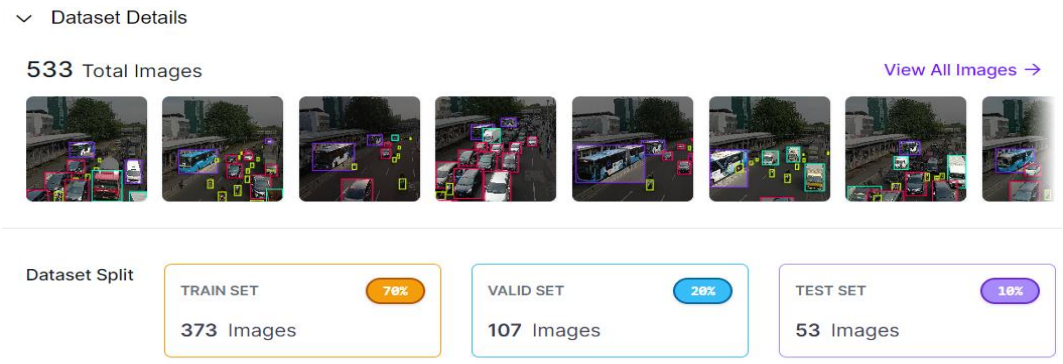
Gambar 3.7 merupakan ilustrasi dari langkah – langkah secara teknis sebagai panduan penulis untuk membuat sistem deteksi objek menggunakan YOLO.

3.2.7 *Setup Environment*

Sebelum melakukan tahap selanjutnya, terdapat beberapa *requirement* yang diperlukan untuk model YOLO ini, dimana akan dilakukan proses instalasi terlebih dahulu seperti *Roboflow*. Tahap perancangan ini dilakukan menggunakan *Google Collab.*

3.2.8 Pengolahan *Dataset*

Dataset yang dimanfaatkan dalam penelitian ini diperoleh dari hasil pengambilan gambar secara pribadi oleh penulis yang dilakukan di Kawasan Jelambar, Jakarta Barat dengan mengambil gambar setiap ada *bus* TransJakarta yang melintas bersamaan dengan kendaraan lain yang juga melintas. *Dataset* terdiri dari kendaraan bus TransJakarta, Truk, Sepeda Motor, Mobil. Gambar 3.5 merupakan *dataset* yang dipergunakan dalam penelitian ini.



Gambar 3.8 Dataset Penelitian

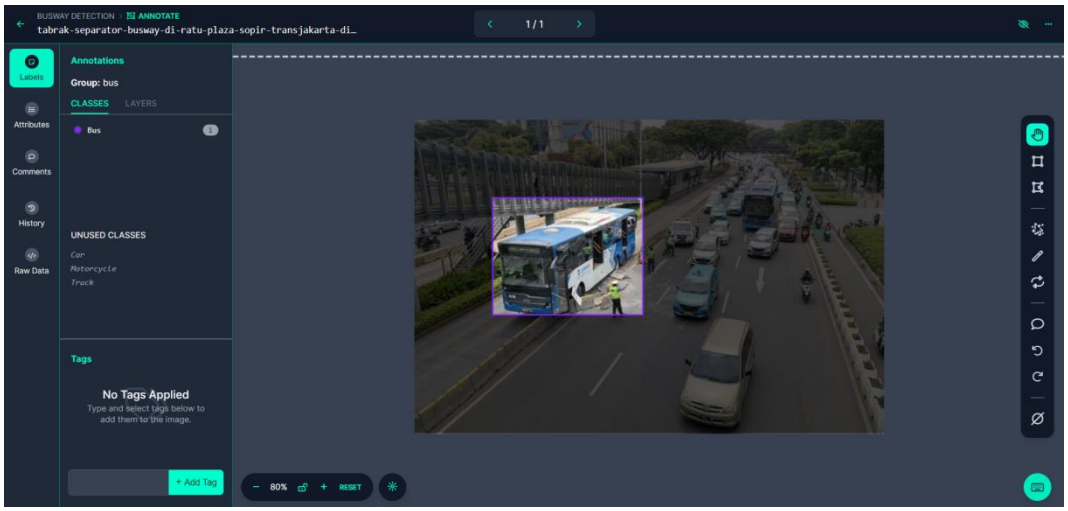
Gambar 3.8 menggambarkan bagaimana dataset dibagi, di mana 70% dari total gambar digunakan sebagai data pelatihan, 20% digunakan untuk validasi, dan 10% dari data pelatihan digunakan sebagai data uji.

Gambar 3.9 Kondisi Lalu Lintas (Truk, Mobil, Motor, TransJakarta)

Gambar 3.9 menunjukkan kondisi lalu lintas yang dijadikan dataset untuk deteksi objek. Pada gambar tersebut terdiri dari beberapa kendaraan diantaranya bus TransJakarta, Mobil, Sepeda Motor, dan Truk.

3.2.9 Labeling Dataset

Labeling dataset YOLO *(You Only Look Once)* anotasi pada setiap objek dalam sebuah dataset menggunakan format YOLO. Dalam konteks *labeling dataset* YOLO, setiap objek pada gambar akan diberikan label sesuai dengan kelasnya masing – masing, serta kotak pembatas *(bounding box)* yang mengelilingi objek tersebut.



Gambar 3.10 Proses Labeling Dataset

Pada Gambar 3.10 adalah proses melakukan pelabelan terhadap objek yang akan dideteksi, pada penelitian ini untuk proses pelabelan dilakukan menggunakan API Roboflow. Untuk proses pelabelannya dengan memilih menu *annotate* kemudian pilih gambar yang akan dilabelkan kemudian untuk label yang digunakan terdiri dari berbagai macam bentuk terdapat bentuk kotak, polygon, dan sebagainya.

3.2.10 *Training Dataset*

Setelah dataset diberikan label, kemudian di-*training* untuk dapat digunakan dalam mendeteksi objek. Training ini dilakukan menggunakan algoritma YOLO yang metodenya berbasis CNN. Pada prosesnya, data *training* diambil menggunakan Google Colaboratory. Setelah *training* selesai, maka perlu evaluasi kinerja model menggunakan data validasi atau data uji yang terpisah. Dapat juga melihat nilai metrik seperti presisi, *recall.*