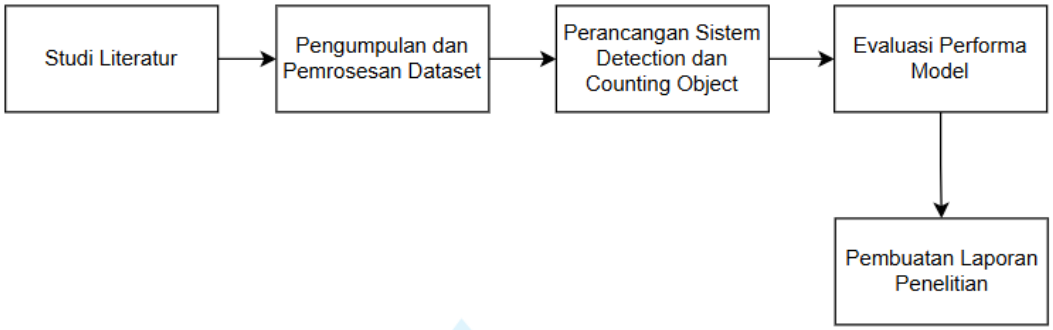
BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Pada penelitian ini, terdapat 5 tahap penelitian yang dilakukan oleh penulis untuk mendapatkan hasil penelitian yaitu dengan studi *literatur*e, pengumpulan dan pemrosesan *dataset*, perancangan sistem deteksi dan hitung otomatis pada *object*, melakukan evaluasi metrik pada performa model, dan pembuatan laporan penelitian. Pada Gambar 3.1 dapat dilihat untuk alur metode penelitian lebih jelas.



Gambar 3. 1 Diagram Tahapan Penelitian

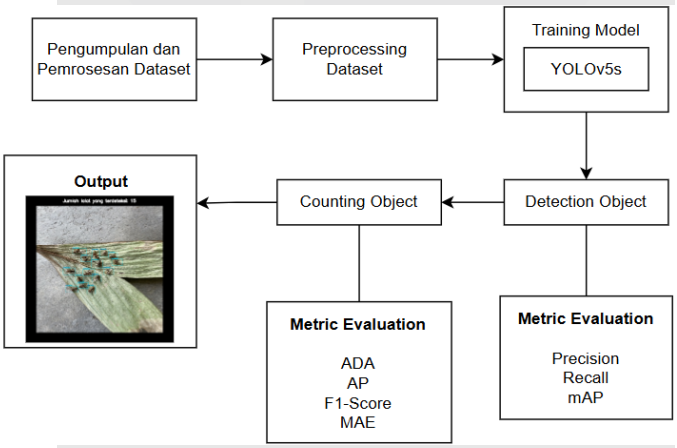
3.2 Studi *Literature*

Studi *literature* yang dilakukan oleh penulis adalah mencari informasi berupa penelitian terdahulu dan dokumentasi teknologi yang berkaitan dan mendukung penelitian ini. Dalam hal ini adalah penelitian yang terkait dengan *deep learning* untuk deteksi dan *counting* objek, model YOLOv5, dan *metric evaluation* untuk deteksi dan hitung objek. Setelah menemukan penelitian terkait, maka dianalisis pada spesifikasi sistem yang akan dibangun, implementasi sistem, dan metode untuk mendapatkan hasil pada penelitian. Penulis juga melakukan pencairan terhadap definisi lalat buah salak untuk mengetahui karakteristik *object* yang digunakan dalam penelitian. Penulis juga melakukan bimbingan kepada dosen Program Studi Teknik Komputer untuk mendapatkan informasi terkait penelitian

yang dilakukan. Tujuan tahap ini adalah untuk memberikan informasi dan referensi yang cukup memadai dalam penulisan penelitian yang akan dirancang dan dibangun.

3.3 Arsitektur Sistem Deteksi dan Hitung Otomatis Lalat Buah

Sebelum merancang sistem, Gambar 3.2 memberikan gambaran awal tentang perancangan sistem deteksi dan penghitungan otomatis objek lalat buah pada salak menggunakan model YOLOv5. Gambar ini menjadi landasan dalam pemilihan modul sistem yang akan digunakan, serta disesuaikan dengan kebutuhan, metode, arsitektur, dan elemen lainnya. Dalam sistem deteksi dan penghitungan otomatis ini, digunakan *dataset* primer yang telah dianotasi. Proses *preprocessing dataset* dilakukan untuk mempersiapkan *dataset*, termasuk menyamakan resolusi, melakukan pengaturan *dataset* gambar guna menghindari *overfitting*. Setelah model dilatih, model tersebut dievaluasi, dan jika performanya baik, proses *counting object* dilanjutkan. Evaluasi juga dilakukan pada tahap *counting object* untuk memastikan kinerja model sesuai dengan yang diharapkan.

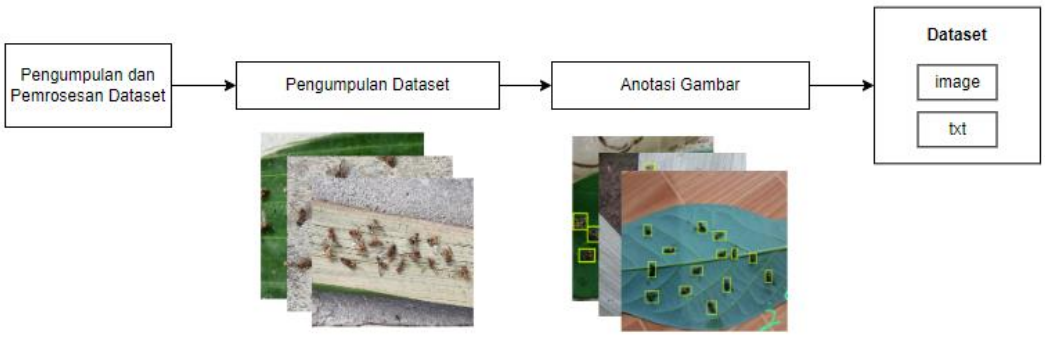


Gambar 3. 2 Diagram Arsitektur Sistem secara General

3.4 Pengumpulan dan Pemrosesan Dataset

Pengumpulan dan pemrosesan *dataset* merupakan fondasi penting dalam pengembangan sistem deteksi dan penghitungan otomatis objek lalat buah salak. *Dataset* yang digunakan dalam penelitian ini harus dikumpulkan secara tepat dan

beragam untuk meningkatkan variasi data yang tersedia, sehingga model dapat belajar dari berbagai kondisi. Pemrosesan *dataset* juga memainkan peran penting dalam memastikan kesiapan data sebelum digunakan untuk melatih model, yang bertujuan untuk mencapai akurasi tinggi. Langkah-langkah pemrosesan seperti anotasi gambar dilakukan untuk mempersiapkan data sesuai dengan kebutuhan model YOLOv5. Hasil dari pengumpulan dan pemrosesan ini menghasilkan kumpulan *dataset* yang terdiri dari gambar dan label dalam format teks, seperti yang ditunjukkan pada Gambar 3.3. Informasi ini kemudian digunakan oleh model YOLOv5 untuk mempelajari *dataset* dengan baik, sehingga dapat mencapai performa yang optimal dalam deteksi dan penghitungan objek.



Gambar 3. 3 Alur Pengumpulan dan Pemrosesan Dataset

3.4.1 Pengumpulan *Dataset*

- Dataset Internal

  Dataset internal dikumpulkan secara langsung selama satu minggu dengan mengunjungi lahan pertanian kelompok tani salak Mitra Turindo, menghasilkan total 6.008 gambar.

- Dataset Eksternal

  Dataset eksternal diperoleh dari foto-foto laporan tangkapan petani yang dikumpulkan setiap minggu selama periode Maret hingga Desember, dengan total sebanyak 148 gambar. Setiap gambar yang dikumpulkan telah divalidasi oleh ahli pertanian. Validasi ini bertujuan untuk

memastikan akurasi dan relevansi data dalam konteks identifikasi dan penghitungan lalat buah.

Foto lalat buah dikumpulkan dari hasil perangkap yang dibuat oleh petani. Perangkap ini dipasang di sekitar lahan perkebunan petani. Lalat yang tertangkap di perangkap kemudian ditebar dan ditata dengan sedikit berjarak di atas media, seperti pada Gambar 3.4. Media yang biasanya digunakan seperti ubin, daun, kayu, atau lainnya. Pemilihan media ini tidak hanya mempertimbangkan visibilitas lalat buah dalam gambar, tetapi juga kemudahan akses dan ketersediaan di lokasi pengumpulan data.

Tabel 3. 1 Perangkat yang Digunakan

Perangkat	Resolusi Gambar
Iphone 12 Mini	3024 x 4032
Iphone 12 Pro	3024 x 4032
Oppo reno7 z 5g	3468 x 4624
Redmi K20	2448 x 3264
Xiaomi 11T	2233 x 3968
Samsung S23 fe	3000x4000
Sony Cybershot 16	1920 x 1080 640 x 480

Setelah penyusunan lalat buah di atas media, gambar diambil menggunakan kamera yang bervariasi agar dataset yang dikumpulkan menjadi beragam, seperti yang ditunjukkan pada Tabel 3.1. Pada Gambar 3.5, dapat dilihat berbagai pergantian media yang digunakan dalam pengumpulan dataset, yaitu daun kering (a), ubin (b), dan daun (c). Selain itu, juga mengumpulkan dataset dalam berbagai kondisi lalat buah, seperti kondisi kering (a,b,c) dan basah (d), seperti yang ditunjukkan pada Gambar 3.5. Gambar diambil dengan variasi posisi dan sudut pengambilan gambar untuk memastikan keberagaman dataset yang dikumpulkan, seperti yang ditunjukkan pada

Gambar 3.6. Ini mencakup variasi dalam ukuran, posisi, dan jumlah lalat buah.

Hal ini bertujuan untuk menambah variasi dalam dataset, yang penting agar model YOLOv5 yang dilatih dengan dataset tersebut dapat mengenali dan menghitung lalat buah dengan akurasi tinggi, bahkan dalam situasi yang kompleks di lapangan. Variasi dalam resolusi gambar, seperti yang ditunjukkan dalam Tabel 3.1, turut membantu model dalam mengenali berbagai kondisi dan memastikan deteksi yang lebih andal dalam berbagai lingkungan dan kondisi pengambilan gambar.

Gambar 3. 4 Proses Pengambilan Dataset

(a) (b) (c) (d)

Gambar 3. 5 Dataset Lalat Buah

Gambar 3. 6 Keberagaman Dataset

3.4.2 Anotasi Gambar

Anotasi gambar menjadi proses penting dalam pemrosesan *dataset* dengan setiap objek yang relevan dalam gambar seperti lalat buah akan diidentifikasi dan diberi label secara manual. Proses ini melibatkan penggambaran kotak pembatas (*bounding boxes*) di sekitar objek yang ingin dideteksi oleh model, serta penambahan label yang menunjukkan jenis objek tersebut. Proses Anotasi gambar dilakukan menggunakan *framework Roboflow* [30]. Proses anotasi ini sangat krusial karena kualitas anotasi langsung mempengaruhi kinerja model yang akan dilatih. Anotasi yang akurat membantu model untuk belajar membedakan antara objek yang diinginkan (lalat buah) dan elemen latar belakang atau objek lain yang tidak relevan. Pada proses anotasi *dataset* lalat buah, terdapat syarat khusus yang harus dipenuhi: tidak hanya lalat buah yang utuh yang dideteksi sebagai lalat buah, tetapi bagian toraks lalat buah juga diidentifikasi sebagai objek yang relevan, seperti yang ditunjukkan pada Gambar 3.7. Namun, jika dalam gambar hanya terlihat bagian kepala atau abdomen (perut) lalat buah, bagian tersebut tidak akan diidentifikasi sebagai lalat buah. Hal ini disebabkan karena bagian tubuh tersebut mungkin saja terbawa oleh lalat buah lain yang masuk ke perangkap, sehingga tidak dapat diidentifikasi secara akurat sebagai individu lalat buah yang terpisah. Pendekatan ini diterapkan untuk meningkatkan akurasi model dalam mendeteksi dan menghitung jumlah lalat buah secara tepat, dengan meminimalkan kesalahan yang disebabkan oleh bagian tubuh yang terlepas atau tidak utuh. Kasus seperti ini jarang terjadi dan tidak memberikan dampak signifikan pada performa model.

Selain label "lalat", anotasi gambar juga mencakup label "0". Label "0" digunakan untuk kondisi di mana lalat bergerombol dalam jumlah besar sehingga sulit diidentifikasi satu per satu. Contoh kondisi ini dapat dilihat pada Gambar 3.8, yang menunjukkan situasi di mana label "0" diterapkan. Total label dalam dataset terdiri dari 5241 label "0" dan 141891 label "lalat".

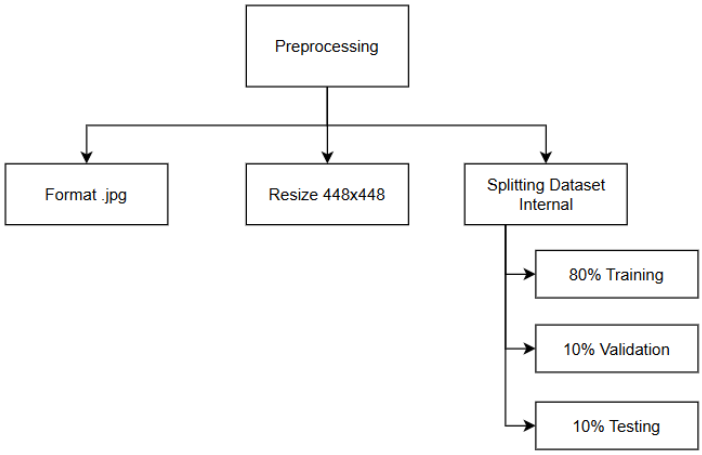
Gambar 3. 7 Bagian Toraks yang Diidentifikasi sebagai Lalat Buah

Gambar 3. 8 Dataset dengan Label 0

3.5 Preprocessing

Sebelum memulai pelatihan *dataset* pada model YOLOv5, ada beberapa langkah penting yang harus dilakukan dalam proses *preprocessing* data seperti yang

ditunjukkan pada Gambar 3.9 untuk memastikan bahwa *dataset* siap digunakan dan dapat diproses oleh model dengan efisien. *Dataset* yang dikumpulkan berasal dari kamera dengan resolusi yang berbeda-beda, sehingga langkah pertama dalam *preprocessing* adalah melakukan *resize* gambar. Ukuran gambar diubah menjadi 448x448 piksel, yang dipilih agar sesuai dengan kebutuhan model. Ukuran ini cukup besar untuk mempertahankan detail objek yang penting seperti lalat buah, namun tidak terlalu besar sehingga membuat komputasi model menjadi lebih ringan. Selain *resize,* semua gambar dalam *dataset* juga perlu diubah ke format JPG karena gambar diambil dengan jenis kamera yang berbeda-beda yaitu 3024 x 4032, 3468 x 4624, 2448 x 3264, 2233 x 3968, 3000 x 4000, 1920 x 1080, dan 640 x 480. Selanjutnya, *dataset* internal dibagi menjadi tiga bagian dengan skala 80:10:10, yaitu 80% untuk *training*, 10% untuk validasi, dan 10% untuk *testing* internal.



Gambar 3. 9 *Preprocessing* Dataset

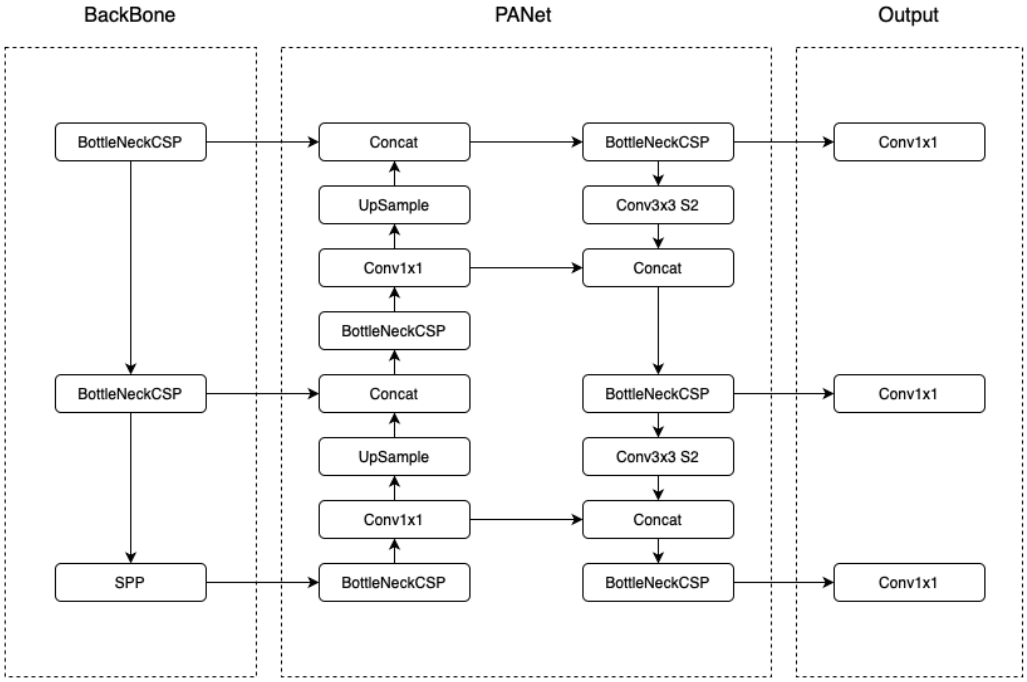
3.6 *Training* Model YOLOv5

Setelah pengumpulan *dataset*, anotasi gambar, dan *preprocessing*, langkah selanjutnya adalah melatih model YOLOv5 dengan menggunakan *dataset* internal *training* dan validasi untuk mendeteksi dan menghitung objek lalat buah. *Training* akan membahas secara detail mengenai arsitektur yang digunakan pada YOLOv5 dan pengaturan parameter untuk memastikan model dapat belajar dengan optimal.

Proses pelatihan ini merupakan inti dari perancangan sistem deteksi dan hitung otomatis, karena model akan belajar mengenali pola dari data yang telah dipersiapkan dan mengaplikasikannya dalam tugas deteksi dan *counting* pada objek.

3.6.1 Arsitektur YOLOv5

Algoritma YOLOv5 terdiri dari tiga modul yaitu *CSP-DarkNet backbone*, FPN+PAN *neck*, dan *prediction head*. Seperti yang ditunjukan pada Gambar 3.10, gambar dengan ukuran 448 x 448 x 3 dimasukkan ke dalam jaringan. Pada bagian *backbone*, lapisan CBS digunakan untuk *downsampling*, dan modul CSP digunakan untuk ekstraksi fitur. Setelah 5 kali *downsampling*, ukuran peta fitur menjadi 512×20×20. Terakhir, modul SPPF dihubungkan untuk mewujudkan penggabungan peta fitur dari bidang reseptif yang berbeda. Pada bagian jaringan *neck*, peta fitur pertama-tama melewati jalur pengurangan dimensi, dan kemudian melalui jalur peningkatan dimensi. Peta fitur dengan ukuran 512 × 20 × 20, 256 × 40 × 40, dan 128 × 80 × 80 digabungkan sepenuhnya melalui dua jalur. Pada bagian jaringan *head*, peta fitur dengan tiga ukuran memasuki kepala pendeteksian dan kemudian melewati lapisan konvolusi 1 × 1. Ukurannya tetap tidak berubah, dan jumlah saluran menjadi 3 × (NC + 5), di mana 3 mewakili tiga jenis kotak jangkar dengan rasio aspek yang berbeda, NC mewakili jumlah kategori, dan 5 mewakili 4 parameter yang digunakan untuk mengindikasikan posisi bingkai jangkar ditambah 1 probabilitas latar depan bingkai jangkar [31].



Gambar 3. 10 Arsitektur YOLOv5 [32]

Pada arsitektur YOLOv5 terdapat bagian *CSP-Darknet* yang dirancang untuk mengurangi beban komputasi sehingga meningkatkan efisiensi kinerja model. Efisiensi kinerja model dilakukan dengan membagi *feature map* menjadi dua bagian dan memprosesnya secara paralel tanpa mengurangi hasil akurasi. *CSP-Darknet* memanfaatkan blok *residual* yang ada dalam *Darknet* dengan cara yang lebih optimal. Blok *residual* membantu model dalam menangkap informasi yang lebih dalam dan kompleks tanpa memperlambat proses pelatihan atau inferensi. Hal tersebut memungkinkan YOLOv5 untuk mencapai performa yang lebih baik dengan *resource* yang lebih sedikit dibandingkan dengan arsitektur model yang lain.

3.6.2 Pengaturan Hyperparameter

Setelah melalui tahap pengumpulan dan pemrosesan *dataset*, serta mempersiapkan data yang siap digunakan untuk pelatihan, langkah selanjutnya adalah mengatur *hyperparameter* yang tepat untuk melatih model YOLOv5. Pengaturan

hyperparameter sangat penting karena mempengaruhi bagaimana model mempelajari *dataset* dan dapat menentukan seberapa baik model tersebut beradaptasi dengan data baru. *Training* dilakukan dengan mengganti *hyperparameter* yang digunakan agar dapat membantu model mempelajari *dataset* dengan lebih efektif.

Tabel 3. 2 Pengaturan *Hyperparameter*

Hyperparameter	**Keterangan**
Batch Size	16
Epoch	100
Learning Rate	lr0: 0.01, lrf: 0.2
Early Stop	10
Optimizer	Auto

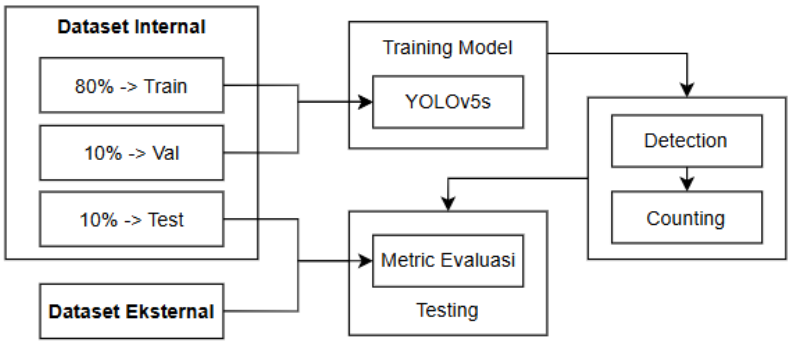
Pada Tabel 3.2 dapat dilihat bahwa pengaturan *batch size* sebesar 16 untuk menghindari penggunaan memori GPU yang berlebihan dan membuat model melakukan *update* parameter lebih sering karena banyaknya sampel 16 digunakan dalam satu kali iterasi *training*. Jumlah *epoch* 100 memberikan waktu yang cukup bagi model untuk konvergen. Dikombinasikan dengan *early stopping* 10, ini memungkinkan model untuk berhenti lebih awal jika tidak ada peningkatan. *Learning rate* awal (lr0) 0.01 merupakan nilai yang cukup konservatif untuk memulai *training*. *Learning rate* final (lrf) 0.2 menunjukkan penggunaan *learning rate scheduler* yang akan menurunkan *learning rate* secara bertahap. Kombinasi ini membantu model mencapai konvergensi yang lebih baik dan menghindari osilasi pada *loss*. Penggunaan *optimizer* 'Auto' menunjukkan sistem akan memilih *optimizer* yang paling sesuai secara otomatis sehingga mengurangi kompleksitas dalam pemilihan *optimizer* secara manual.

Pada proses pelatihan model menggunakan teknik *tranfer learning* yaitu dengan bobot awal dari varian YOLOv5s, yang dikenal sebagai model yang lebih ringan

dan cepat, tetapi tetap mempertahankan akurasi yang baik. Hasil dari pelatihan ini diharapkan menghasilkan model yang tidak hanya akurat dalam mendeteksi dan menghitung objek, tetapi juga efisien dalam pengimplementasiannya.

3.7 Metric Evaluation

Kinerja model diukur dan dievaluasi melalui metrik evaluasi yang digunakan untuk menilai seberapa baik model dalam mendeteksi dan menghitung objek dengan skema seperti Gambar 3.11 di bawah ini.



Gambar 3. 11 Skema Pengujian Model

Metrik evaluasi merupakan alat yang penting untuk mengevaluasi performa model secara objektif, memastikan bahwa model tidak hanya bekerja dengan baik pada data pelatihan, tetapi juga dapat diandalkan saat diimplementasikan. Metrik evaluasi akan diuji dengan menggunakan 2 *dataset* yaitu set *test* (*dataset* internal) dan *dataset* eksternal.

3.7.1 *Metric Evaluation* untuk Deteksi Objek

Evaluasi metrik perlu dilakukan untuk mengevaluasi kinerja dari model Evaluasi metrik dilakukan dengan membandingkan indikator nilai yang dihasilkan seperti *precision*, *recall*, dan mAP. *Precision* adalah perbandingan nilai prediksi benar positif dibandingkan dengan total hasil dengan prediksi positif. *Recall* adalah perbandingan nilai prediksi benar positif dengan seluruh data yang benar positif. *Mean Average*

Precision (mAP) yang merupakan metrik yang digunakan untuk mengukur akurasi dari deteksi objek. Nilai *Precision*, *Recall*, dan mAP 0.5 dapat dicari menggunakan Persamaan (1), (2), dan (3).

$$Precision = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \quad (1)$$

$$Recall = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \quad (2)$$

$$\text{mAP } 0.5 = \frac{1}{N} \sum_{i=1}^{N} APi \quad (3)$$

3.7.2 Metric Evaluation untuk Counting Objek

Metrik evaluasi yang digunakan untuk mengukur kinerja sistem deteksi dan penghitungan objek yaitu ADA, AP, F1-Score, dan MAE. ADA (*Average Detection Accuracy*) mengukur rata-rata akurasi deteksi objek dengan membandingkan jumlah objek yang dideteksi dengan benar (AC) terhadap jumlah total objek yang sebenarnya ada (HC) ditambah dengan kesalahan deteksi (EM). AP mengukur seberapa baik model dalam mendeteksi objek yang benar di antara semua deteksi yang dilakukan, termasuk kesalahan deteksi (CE). F1-score adalah metrik harmonik yang menggabungkan *Average Precision* (AP) dan *Average Detection Accuracy* (ADA) untuk memberikan gambaran yang lebih seimbang tentang kinerja model. MAE mengukur rata-rata kesalahan absolut antara jumlah objek yang sebenarnya ada (HC) dan jumlah objek yang dideteksi (AC) oleh model. Nilai ADA, AP, F1-Score, dan MAE dapat dicari menggunakan Persamaan (1), (2), (3), dan (4).

Average Detection Accuracy (ADA) (1)

$$\frac{\sum AC}{\sum HC + EM}$$

Average Precision (AP) (2)

$$\frac{\sum AC}{\sum HC + EM + \sum CE}$$

F1-Score (3)

$$\frac{2 \times AP \times ADA}{AP + ADA}$$

Mean Absolute Error (MAE) (4)

$$\frac{1}{N}\sum_{i=1}^{N} |HC - AC|$$

Error Margin (EM) (5)

$$5\% \times HC$$

Kesalahan Hitungan (CE) (6)

$$HC - AC$$

Keterangan:

AC = *Algorithm Count*

HC = *Human Count*

EM = *Error Margin*

CE = Kesalahan Hitungan

N = Jumlah Total Gambar yang Diuji

Eror margin biasanya bernilai 5% dari total perhitungan manusia untuk seluruh gambar yang dilakukan pengujian (Z_2).