BAB II
TINJAUAN PUSTAKA

Bab ini akan mencakup pencapaian terdahulu dan tinjauan teoritis yang bertujuan untuk memperkuat penelitian yang dilakukan oleh peneliti.

2.1 Pencapaian Terdahulu

Pencapaian terdahulu berfungsi sebagai referensi yang akan digunakan oleh peneliti dalam memperkuat argument dan sebagai acuan dalam melakukan penelitian. Tidak hanya itu, bab ini juga berfungsi untuk membantu peniliti dalam menghindari duplikasi pada penilitian yang sudah dilakukan sebelumnya dan membantu peniliti dalam membuktikan relevansi antara fenomena dan algoritma yang akan diterapkan. Berikut merupakan tabel 2.1 yang mencakup referensi terdahulu yang berisikan publikasi ilmiah/jurnal terkait penelitian yang akan dilakukan.

a) Perancangan Alat Identifikasi Wajah Dengan Algoritma You Only Look Once (YOLO) Untuk Presensi Mahasiswa

   Penelitian yang dilakukan oleh (Salamah, Said, & Soim, 2022). Hasil penelitian tersebut menunjukkan tingkat akurasi rata-rata sebesar 0,9793. Keberhasilan ini menunjukkan bahwa algoritma YOLO dapat mengatasi tantangan deteksi wajah dengan baik, bahkan dalam kondisi variasi pencahayaan yang berbeda dan waktu yang real-time dalam mengirimkan data ke *website* monitoring presensi mahasiswa. Diharapkan, pengaplikasian teknologi ini akan memungkinkan proses absensi menjadi lebih efektif, efisien, dan transparan, serta memberikan kemudahan dalam pemantauan oleh para dosen. Selain itu, penelitian ini juga memberikan sumbangan positif dalam menghadapi era digitalisasi yang semakin berkembang di bidang pendidikan.

b) Systematic Literature Review Mendeteksi Wajah Manusia Menggunakan Metode YOLO (You Only Look Once)

   Penelitian yang dilakukan oleh (Baihaqi, Firliansyah, Jaelani, Jaya, & Rosyani, 2023). Detail hasil penelitian tersebut Nilai akurasi yang diperoleh adalah Precision 0.253, Recall 0.247, dan F1-Score 0.25, yang dipengaruhi

oleh model CNN dan keseimbangan data target. Terakhir, pengujian pengenalan wajah dan GPS menggunakan YOLO pada platform Android menghasilkan akurasi sebesar 0.93435 dalam menghadapi 20 data penilaian mahasiswa. Hasil ini mengindikasikan potensi YOLO dalam aplikasi deteksi wajah dalam berbagai konteks dan situasi.

c) ANALISIS DETEKSI OBJEK CITRA DIGITAL MENGGUNAKAN ALGORITMA YOLO DAN CNN DENGAN ARSITEKTUR REPVGG PADA SISTEM PENDETEKSIAN DAN PENGENALAN EKSPRESI WAJAH

Penelitian yang dilakukan oleh (Saputri, Taqwa, & Soim, 2022). Hasil pengujian menunjukkan bahwa algoritma YOLO mampu mendeteksi ekspresi wajah dengan tingkat keyakinan sebesar 89%, sementara sistem secara keseluruhan mengklasifikasikan ekspresi senang(*happy)* dengan tingkat keyakinan sebesar 99,3%. Penggunaan metode ini secara bersama-sama memberikan hasil yang efektif dalam mengenali dan mengklasifikasikan ekspresi wajah, dengan potensi aplikasi yang dapat meningkatkan pengenalan emosi wajah manusia dalam berbagai konteks.

d) DETEKSI WAJAH MANUSIA BERBASIS ONE STAGE DETECTOR MENGGUNAKAN METODE YOU ONLY LOOK ONCE (YOLO)

Penelitian yang dilakukan oleh (Thoriq, Siradjuddin, & Permana, 2023). Hasil penelitian ini mengimplementasikan pendekatan deteksi wajah menggunakan algoritma You Only Look Once (YOLO) dengan arsitektur Convolutional Neural Network (CNN) berbasis model VGG-16 untuk perbaikan lokasi dan klasifikasi wajah. Pendekatan one stage detector tersebut membagi data citra menjadi grid dengan ukuran tertentu untuk merepresentasikan lokasi kandidat wajah. Penggunaan arsitektur CNN, terutama model VGG-16, pada lapisan konvolusi serta proses pelatihan pada lapisan Fully Connected (FC) memberikan hasil akurasi. Ujicoba dilakukan pada dataset WIDER Face dengan variasi jumlah wajah dalam setiap citra. Hasil ujicoba menunjukkan nilai akurasi Precision sebesar 0.253, nilai akurasi Recall sebesar 0.247, dan nilai akurasi F1-Score sebesar 0.25. Dengan demikian, metode deteksi wajah yang diusulkan telah berhasil

diimplementasikan dan diuji pada dataset yang relevan, dengan akurasi yang dapat memberikan kontribusi dalam aplikasi pengenalan wajah dan pencarian wajah dalam citra.

e) Pemanfaatan Google Colab Untuk Aplikasi Pendeteksian Masker Wajah Menggunakan Algoritma Deep Learning YOLOv7

Penelitian yang dilakukan oleh (Guntara, 2023). Hasil dari penelitian ini menunjukkan bahwa deteksi masker wajah menggunakan metode deteksi objek *Deep Learning* memiliki peran penting dalam upaya melawan COVID-19. Metode ini telah mencapai hasil yang baik dengan tingkat akurasi yang tinggi. Dalam tinjauan yang komprehensif terhadap teknik deteksi masker wajah, teridentifikasi beberapa algoritma berbasis deep learning, termasuk *You Only Look Once* (YOLO), *Single Shot Detector* (SSD), *RetinaFace*, dan *Faster* R-CNN. Sebelumnya, studi-studi fokus pada akurasi deteksi masker wajah menggunakan model deteksi dua tahap seperti Faster R-CNN. Namun, dalam penelitian ini, perhatian khusus diberikan pada detektor satu tahap, yaitu YOLO. Hasil training dalam penelitian ini menunjukkan bahwa nilai *Precision* (presisi) tetap konsisten dalam kisaran 0,4 hingga 0,8. Nilai *Recall* (sensitivitas) maksimum mencapai 0,6.

f) Deteksi Penggunaan Masker Wajah Menggunakan YOLOv5

Penelitian yang dilakukan oleh (Dawami, Rachmawati, & Sulistyo, 1746-1764). Hasil penelitian menunjukkan bahwa dalam menggunakan dataset deteksi masker wajah yang asli, nilai IoU (*Intersection over Union*) threshold terbaik adalah 0,3, yang menghasilkan nilai mean *Average* Precision (mAP) sebesar 0,876 saat pengujian untuk semua kelas. Sementara itu, jika menggunakan dataset deteksi masker wajah yang telah diaugmentasi, nilai IoU *threshold* terbaik adalah 0,5, dengan mAP sebesar 0,849 saat pengujian untuk semua kelas. Dengan demikian, penelitian ini menunjukkan bahwa sistem deteksi menggunakan metode YOLOv5 memiliki potensi untuk mendeteksi penggunaan masker wajah pada manusia dengan akurasi yang baik. Penggunaan dataset asli atau yang telah diaugmentasi memengaruhi nilai IoU *threshold* terbaik yang diperlukan untuk mencapai tingkat mAP tertinggi. Hasil ini memiliki implikasi penting

dalam memastikan disiplin masyarakat dalam menjalankan protokol kesehatan pandemi COVID-19 dengan lebih baik.

2.2 Tinjauan Teoritis

Tinjauan teoritis dalam penelitian ini berfungsi sebagai referensi dan pendukung informasi dalam memperkuat justifikasi penelitian. Adapun penjelasan detail mengenai tinjauan teoritis yang digunakan dalam penelitian ini, dijabarkan sebagai berikut.

2.2.1 *Artificial Intelligence*

Artificial Intelligence (AI) adalah cabang ilmu komputer yang berfokus pada pengembangan mesin atau sistem komputer dan dapat melakukan tugas yang membutuhkan kecerdasan manusia. Tujuannya adalah untuk menciptakan mesin yang dapat belajar, berpikir, dan bertindak seperti manusia. AI mencakup berbagai teknik dan metode, seperti pemrograman logika, pemrosesan bahasa alami, dan pengenalan pola. Dalam pemrograman logika, sistem AI menggunakan aturan logika untuk membuat keputusan berdasarkan fakta-fakta yang diberikan. Pemrosesan bahasa alami memungkinkan sistem AI untuk memahami dan menghasilkan bahasa manusia.

a. *Machine Learning* dan *Deep Learning*

Machine learning adalah cabang dari kecerdasan buatan (*artificial intelligence*) yang berfokus pada pengembangan algoritma dan model komputer yang memungkinkan sistem untuk belajar dari data dan pengalaman, serta meningkatkan kinerjanya dalam tugas tertentu tanpa perlu secara eksplisit diprogram (Fathurohman, 2021). Algoritma dan model dalam *machine learning* dikembangkan untuk mengenali pola dalam data dan membuat prediksi atau keputusan berdasarkan pola tersebut. Selain itu, *machine learning* juga digunakan untuk membuat model yang bertujuan untuk menganalisis data gambar atau video.

Seiring dengan kemajuan dalam bidang machine learning, muncul sub-bidang yang dikenal sebagai *Deep Learning. Deep learning* adalah sub bidang dari *machine learning* yang menggunakan jaringan saraf tiruan

(*artificial neural networks*) yang kompleks dan dalam (disebut "*deep*") untuk mengekstraksi sistem dari data (Raup, Ridwan, Khoeriyah, Supiana, & Zaqiah, 2022). *Deep learning* memungkinkan komputer untuk belajar secara hierarkis dan otomatis dari data yang sangat besar dan kompleks, seperti gambar, suara, teks, dan lainnya menggunakan bantuan jaringan syaraf tiruan (Syafa'ah & Lestandy, 2021). Ini mencoba untuk mensimulasikan cara kerja otak manusia dengan menggunakan banyak lapisan neuron buatan yang terhubung dalam jaringan untuk menganalisis dan menginterpretasi data.

b. Visi Komputer

Visi komputer merupakan dari bagian dari *deep learning* yang tercakup dalam *machine learning* dengan fokus pada pengembangan algoritma dan model komputer untuk melakukan pemrosesan serta pengolahan gambar (Narhan, 2023). Tujuan utama dari visi komputer adalah membuat komputer mampu "melihat" dan memahami informasi dari gambar atau video, mirip dengan cara manusia mengolah data visual. Visi komputer melibatkan analisis dan ekstraksi fitur dari data visual, seperti pola, warna, tekstur, dan bentuk, dengan menggunakan teknik-teknik seperti pengolahan citra, segmentasi, dan deteksi objek (Ardiansyah, Supit, & Said, 2022).

Algoritma visi komputer biasanya dirancang untuk mengidentifikasi objek, manusia, atau kejadian tertentu dalam gambar atau video, serta untuk mengenali pola-pola yang mungkin sulit dikenali oleh manusia. Penerapan visi komputer dapat ditemukan dalam berbagai bidang, mulai dari pengenalan wajah, deteksi objek dalam kendaraan otonom, hingga pengawasan industri.

Progres dalam visi komputer semakin menarik perhatian akademisi dan praktisi, terutama karena kemampuannya untuk meningkatkan sistem machine learning dalam menghadapi data visual. Secara keseluruhan, visi komputer memberikan kontribusi penting dalam pengembangan teknologi yang dapat memberikan interpretasi yang semakin akurat terhadap dunia visual, membuka potensi baru untuk aplikasi di berbagai industri.

c. Objek

Objek dalam konteks visi komputer merujuk pada entitas visual yang dapat diidentifikasi dan dianalisis oleh algoritma komputer. Pemahaman terhadap objek ini mencakup identifikasi, deteksi, dan pengklasifikasian entitas visual tersebut melalui gambar atau video (Lestari, Putra, Sonita, & Darnita, 2023). Objek sering kali dapat mencakup berbagai jenis entitas, mulai dari manusia, kendaraan, hingga objek lainnya yang memiliki relevansi dalam suatu konteks tertentu.

Proses identifikasi objek melibatkan penggunaan algoritma untuk mengenali karakteristik visual tertentu yang dapat dikaitkan dengan suatu objek. Ini dapat mencakup deteksi tepi, pengenalan pola, dan ekstraksi fitur untuk memahami konten visual dari gambar atau *frame* video. Beberapa pendekatan modern menggunakan teknologi *deep learning*, terutama dengan menggunakan arsitektur jaringan saraf tiruan, untuk meningkatkan akurasi identifikasi objek dalam berbagai kondisi visual.

Deteksi objek melibatkan penentuan lokasi relatif dari objek-objek yang diidentifikasi dalam suatu gambar atau video. Deskripsi pada deteksi tersebut di gambarkan dalam bentuk *bounding box* atau batasan area yang menunjukkan lokasi spasial objek dalam ruang visual. Pengklasifikasian objek kemudian melibatkan pemberian label atau kategori pada objek berdasarkan informasi yang diperoleh dari analisis visual.

Objek dalam visi komputer memiliki peran penting dalam berbagai perangkat lunak, seperti sistem deteksi wajah. Kemampuan untuk mengidentifikasi dan menganalisis objek dalam data visual menjadi kunci untuk meningkatkan kinerja sistem visi komputer dan meningkatkan aplikasi di berbagai industri.

2.2.2 Deteksi Wajah (*Face Detection*)

Deteksi wajah, atau dikenal juga sebagai "*Face Detection*" dalam konteks komputer vision, adalah teknologi yang memanfaatkan algoritma dan analisis citra digital untuk mengenali dan mengekstrak sistem wajah manusia dalam gambar atau video (Diantoro & Gustina, 2019). Tujuan utama dari deteksi wajah adalah untuk mengidentifikasi dan menandai lokasi dari satu atau lebih wajah dalam sebuah

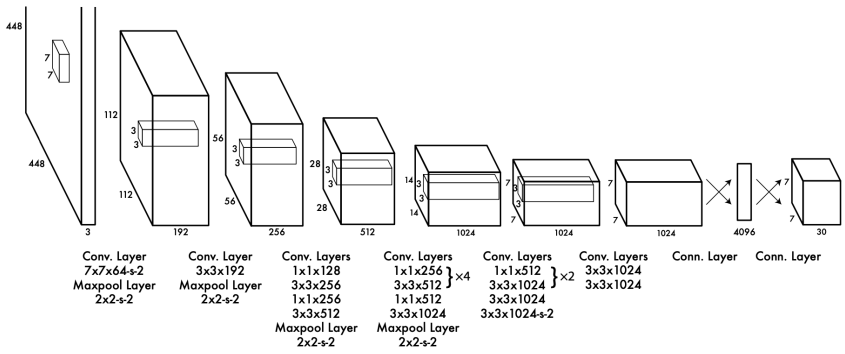
visual, sering kali berupa persegi panjang yang mengelilingi area wajah yang terdeteksi. Proses deteksi wajah melibatkan pengenalan pola dan struktur yang khas dari wajah manusia, seperti mata, hidung, mulut, dan kontur wajah, yang dianalisis oleh algoritma komputer untuk mengenali dan membedakan wajah dari objek atau latar belakang lainnya. Teknologi ini memiliki berbagai aplikasi yang luas, termasuk pengenalan wajah, koreksi gambar, keamanan, *augmented reality*, dan banyak lagi.

2.2.3 YOLO

You Only Look Once (YOLO) adalah salah satu algoritma dalam bidang pengenalan objek dalam gambar atau video secara *real time* (Mulyana & Rofik, 2022). Algoritma ini digunakan untuk mendeteksi objek secara *real-time* dalam gambar atau video dengan kecepatan tinggi (Rahma, Syaputra, Mirza, & Purnamasari, 2021). YOLO berbeda dari pendekatan lain dalam pengenalan objek karena ia mengatasi masalah ini sebagai tugas regresi dari kotak pembatas (*bounding box*) dan kelas objek secara bersamaan dalam satu langkah.

YOLO beroperasi dengan menggambarkan citra yang akan diolah menjadi beberapa jaringan sel. Tahap awal melibatkan pembagian citra menjadi bagian-bagian sel jaringan. Jumlah sel jaringan dapat dihitung dengan membagi resolusi gambar menggunakan langkah (*stride*) yang telah ditetapkan sebelumnya. Tujuannya adalah menghasilkan kotak dengan dimensi 8x8, 16x16, dan 32x32. Dalam pelaksanaan YOLO yang umum, gambar biasanya memiliki resolusi 640x640. Oleh karena itu, langkah yang digunakan adalah 80x80 untuk objek kecil, 40x40 untuk objek sedang, dan 20x20 untuk objek yang lebih besar. YOLO membagi citra input menjadi grid-grid berukuran S x S (Agustina & Sukron, 2022). Setiap grid bertanggung jawab untuk memprediksi keberadaan objek, kotak pembatas, dan tingkat keyakinan (*confidence*). *Confidence* digunakan untuk meramalkan apakah objek ada dalam grid tersebut.

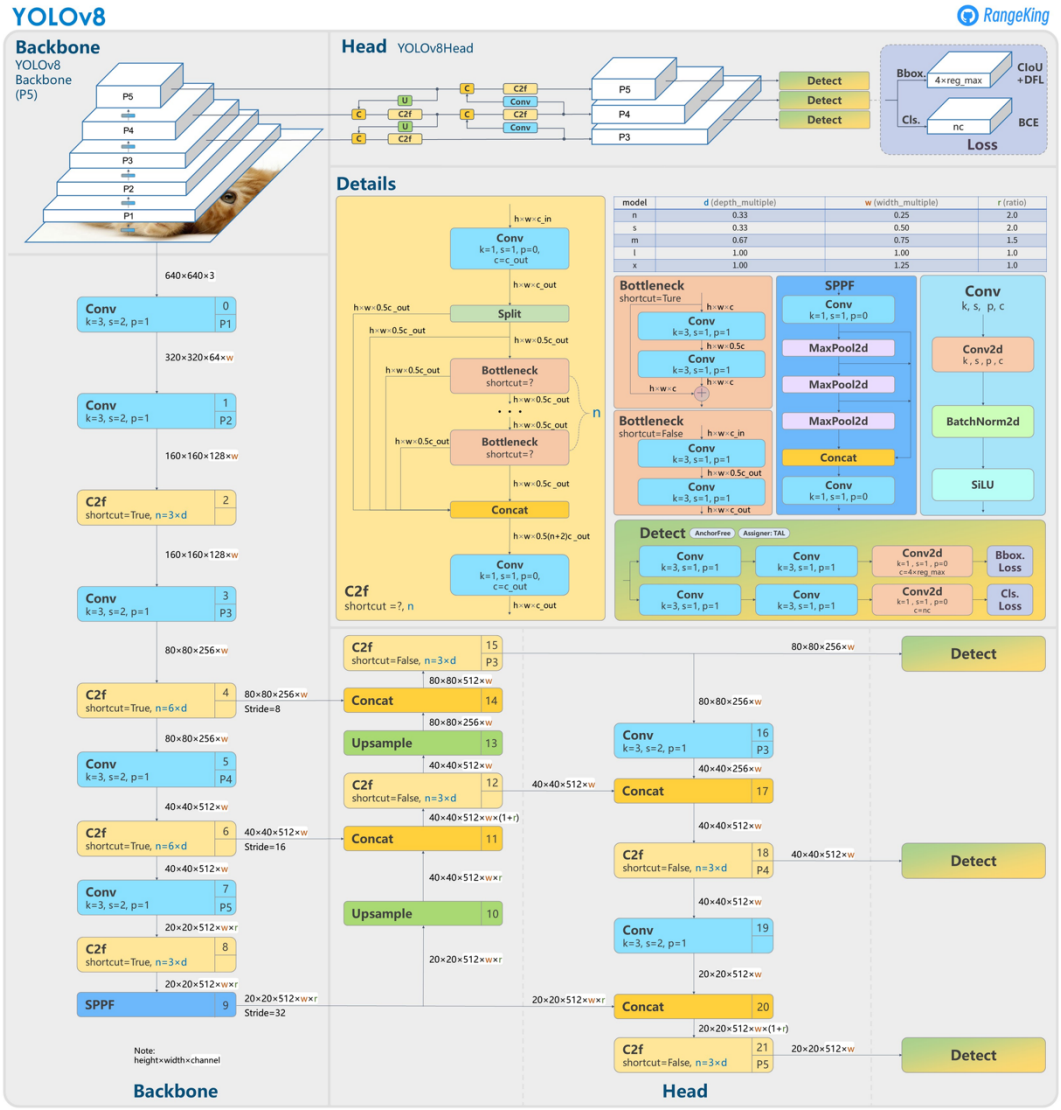
Keuntungan utama dari YOLO adalah kecepatannya. Karena menghindari tahap-tahap terpisah, YOLO dapat memberikan hasil deteksi objek secara real-time dengan kecepatan yang sangat tinggi, membuatnya cocok untuk aplikasi seperti deteksi objek dalam video bergerak atau aplikasi deteksi objek dalam waktu nyata.



Gambar 2.1 Arsitektur YOLO

Gambar 2.1 merupakan detail arsitektur utama algoritma YOLO dalam proses deteksi objek. Masing-masing kotak pembatas memiliki atribut (x, y, w, h, dan *confidence*). Atribut x dan y menunjukkan posisi kotak pembatas, sementara atribut w dan h menggambarkan dimensi tinggi dan lebar kotak pembatas. *Confidence* adalah skor IoU yang dihasilkan dari perbandingan antara kotak prediksi dengan kotak referensi sebenarnya (*ground-truth box*). Pada setiap grid, probabilitas prediksi kelas objek akan dihitung.

Algoritma YOLO memiliki arsitektur tersendiri pada masing-masing versinya. Setiap peningkatan versi pada algoritma tersebut terdapat perubahan pada proses pendeteksian objek yang dilakukan. Arsitektur YOLOv8 terdiri dari beberapa tahap utama ketika melakukan pendeteksian objek, yaitu proses *backbone, bottleneck, head.* Rincian detail arsitektur YOLOv8 tercakup pada gambar di bawah ini.



Gambar 2. 2 Arsitektur YOLOv8

Gambar 2.2 mencakup arsitektur proses pendeteksian yang dilakukan oleh algoritma YOLOv8 dalam mendeteksi objek berdasarkan gambar/citra, bahkan video secara *real-time.* Seluruh proses tersebut dijabarkan sebagai berikut:

a. *Backbone*

Backbone adalah bagian dari arsitektur YOLOv8 yang bertanggung jawab untuk mengekstrak fitur dari gambar input. *Backbone* YOLOv8 menggunakan *DarkNet*-53, yang merupakan model *Convolutional Neural Network* (CNN) yang terdiri dari 53 lapisan. *DarkNet*-53 memiliki kedalaman dan lebar yang lebih besar daripada backbone YOLOv7, sehingga dapat mengekstrak fitur yang lebih detail dari gambar input.

BANGUNAN

Hasil dari *backbone* kemudian dikirim ke *bottleneck* yang terdiri dari beberapa lapisan CNN guna mengurangi dimensi fitur dari *backbone*. Proses ini dilakukan untuk mengurangi beban komputasi yang dibutuhkan oleh *head*.

b. Bottleneck

Bottleneck adalah bagian dari arsitektur YOLOv8 yang menghubungkan *backbone* dengan *head*. *Bottleneck* terdiri dari beberapa lapisan *Convolutional Neural Network* yang berfungsi untuk mengurangi dimensi *output* dari *backbone*. Hal ini dilakukan untuk mengurangi beban komputasi yang dibutuhkan oleh head. Selain itu, terdapat proses *split* pada *bottleneck* yang berfungsi untuk membagi *output* dari layer menjadi dua bagian. Bagian pertama digunakan untuk mengekstrak fitur, sedangkan bagian kedua digunakan untuk mengurangi dimensi fitur. Proses split ini dilakukan untuk mengurangi beban komputasi yang dibutuhkan oleh head. Head membutuhkan fitur yang lebih sedikit daripada backbone.

c. Head

Head adalah bagian dari arsitektur YOLOv8 yang bertanggung jawab untuk menghasilkan *output* deteksi objek. *Head* YOLOv8 terdiri dari tiga bagian, yaitu:

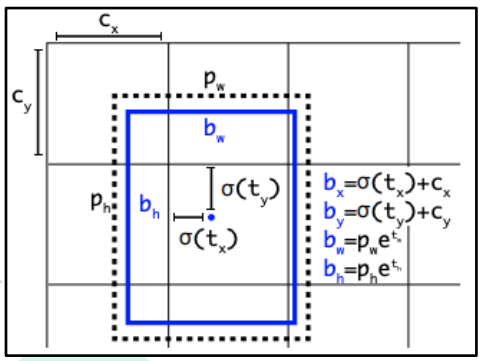
1. Pengekstraksi fitur

   Bagian ini berfungsi untuk mengekstrak fitur dari output *bottleneck*. Fitur dari output *bottleneck* diekstrak dengan menggunakan beberapa layer convolution. Output dari bagian ini memiliki ukuran $320\ x\ 320\ x\ 256$.

2. Detektor *bounding box*

   Bagian ini berfungsi untuk memprediksi *bounding box* dari objek-objek yang ada di gambar input. Proses deteksi *bounding box* pada YOLOv8 dilakukan dengan cara memprediksi empat parameter, yaitu pusat, lebar, tinggi *bounding box* serta probabilitas keberadaan objek. Detektor pada tahap ini akan diprediksi dari beberapa skala untuk meningkatkan akurasi deteksi objek karena skala yang lebih

kecil, model dapat mendeteksi objek yang berukuran kecil, sedangkan pada skala yang lebih besar, model dapat mendeteksi objek yang berukuran besar. Detail penjelasan dalam tahap ini tercakup pada gambar di bawah ini.



Gambar 2. 3 Bounding Box (Anchor Free Detection)

Gambar 2.3 merupakan gambaran *bounding box* yang akan melakukan deteksi pada objek dari input gambar atau video. Pusat *bounding box* adalah titik tengah dari *bounding box*. Lebar *bounding box* adalah jarak antara titik kiri atas dan titik kanan bawah dari *bounding box*. Tinggi *bounding box* adalah jarak antara titik atas dan titik bawah dari *bounding box*. Probabilitas keberadaan objek adalah probabilitas bahwa objek sebenarnya ada di dalam *bounding box* tersebut. Jika dibuat dalam persamaan, maka penjelasan detailnya sebagai berikut:

a. Pusat diprediksi dengan menggunakan operasi *convolution*. *Output* dari operasi *convolution* ini adalah empat angka, yaitu:
   1. *X* : sebagai koordinat X dari pusat *bounding box.*
   2. *Y* : sebagai koordinat Y dari pusat *bounding box.*
   3. *W*: sebagai lebar dari *bounding box.*
   4. *H* : sebagai tinggi dari *bounding box.*

b. Lebar dan tinggi *bounding box* diprediksi dengan menggunakan operasi linear dan memiliki dua otput, yaitu :
   1. *W*: sebagai lebar dari *bounding box.*
   2. *H* : sebagai tinggi dari *bounding box.*

c. Probabilitas keberadaan objek berkisar antara 0 dan 1. Nilai 0 berarti bahwa objek tidak ada di dalam *bounding box*. Nilai 1 berarti bahwa objek pasti ada di dalam *bounding box*.

3. Pengeklasifikasi objek

Bagian ini berfungsi untuk memprediksi kelas dari objek-objek yang ada di gambar input. Kelas dari objek-objek yang ada di gambar input akan diprediksi, misalnya wajah manusia. Proses klasifikasi objek pada YOLOv8 dilakukan dengan cara memprediksi satu label untuk setiap *bounding box*. Label tersebut menunjukkan kategori dari objek yang ada di dalam *bounding box* tersebut. Proses deteksi *bounding box* dan klasifikasi objek pada YOLOv8 dilakukan secara bersamaan guna meningkatkan kinerja deteksi objek.

Setiap gambar yang diproses pada tahap *head* juga akan melalaui proses *unsampling* dan *concat.* Unsampling dilakukan dengan menggunakan operasi upsampling. Operasi upsampling adalah operasi yang digunakan untuk meningkatkan resolusi output dari layer. Operasi ini dilakukan dengan menggunakan *convolutional transpose*. *Convolutional transpose* adalah operasi yang mirip dengan *convolution*, tetapi dilakukan dengan arah yang berlawanan, kemudian dilanjutkan dengan proses *concat* untuk menggabungkan dua *output* dari layer menjadi satu *output*.

2.2.4 Confusion Matrix

Confusion matrix adalah alat/metode pengujian dalam evaluasi kinerja model klasifikasi untuk menggambarkan performa prediksi model pada data yang diketahui dengan benar. Metode ini juga berfungsi untuk menghitung hasil prediksi yang benar dan salah, serta jenis kesalahan yang terjadi. Confusion matrix umumnya terdiri dari empat elemen utama, yaitu:

a) *True Positive* (TP): Jumlah observasi yang benar diprediksi oleh model/sistem sebagai positif.
b) *True Negative* (TN): Jumlah observasi yang benar diprediksi oleh model/sistem sebagai negatif.

c) *False Positive* (FP): Jumlah observasi yang sebenarnya negatif, tetapi salah diprediksi sebagai positif oleh model/sistem (disebut juga "*Type I error*").

d) *False Negative* (FN): Jumlah observasi yang sebenarnya positif, tetapi salah diprediksi sebagai negatif oleh model/sistem (disebut juga "Type II error").

Confusion matrix memungkinkan untuk menghitung berbagai metrik evaluasi, seperti akurasi, presisi, recall, F1-*score*, dan lain-lain, yang memberikan pemahaman yang lebih mendalam tentang kinerja model dalam berbagai konteks (Rahayu, Prianto, & Novia, 2021). *Confusion matrix* sangat berguna dalam mengukur efektivitas dan kelemahan model klasifikasi serta membantu dalam pengambilan keputusan yang lebih baik dalam pengembangan model.

2.2.5.1 Akurasi

Akurasi adalah metode pengujian yang bergantung pada sejauh mana nilai data prediksi sesuai dengan nilai aktualnya (Pratiwi, Handayani, & Sarjana, 2020). Dengan menghitung berapa banyak data yang diklasifikasikan secara benar, kita dapat mengestimasi tingkat akurasi dari prediksi tersebut. Persamaan untuk menghitung akurasi dinyatakan sebagai berikut:

$$Akurasi = \frac{TP+TN}{TP+TN+FP+FN} \qquad (1)$$

2.2.5.2 Presisi

Presisi adalah metode pengujian yang membandingkan jumlah informasi yang relevan yang ditemukan oleh sistem dengan total informasi yang diambil oleh sistem, termasuk yang relevan dan yang tidak relevan. Persamaan untuk menghitung presisi diwujudkan seperti berikut:

$$Presisi = \frac{TP}{TP+FP} \qquad (2)$$

2.2.5.3 *Recall*

Recall adalah suatu metode pengujian yang membandingkan jumlah informasi yang relevan yang berhasil ditemukan oleh sistem dengan total jumlah informasi relevan yang ada dalam kumpulan informasi, termasuk yang ditemukan maupun yang tidak ditemukan oleh sistem. Persamaan untuk menghitung *recall* dapat dijabarkan sebagai berikut:

$$Recall = \frac{TP}{TP+FN} \qquad (3)$$

2.2.5 Asesmen Daring

Asesmen daring, juga dikenal sebagai e-asesmen, mengacu pada proses penilaian atau evaluasi yang dilakukan melalui platform atau perangkat lunak berbasis internet atau teknologi digital lainnya (Masruria, 2021). Dalam asesmen daring, peserta didik atau individu yang dinilai dapat mengakses dan mengikuti ujian, tugas, atau evaluasi lainnya melalui jaringan *online*, tanpa perlu berada di lokasi fisik tertentu.

2.2.7 Sistem Deteksi Wajah

Sistem deteksi wajah adalah sistem komputer atau perangkat lunak yang dirancang untuk mengidentifikasi dan mengenali wajah manusia dalam gambar atau video. Tujuannya adalah untuk mengenali lokasi dan ciri-ciri wajah dalam data visual dan sering digunakan dalam berbagai aplikasi, termasuk keamanan, biometrik, pemantauan, dan pengenalan wajah. Adapun beberapa *tools* yang digunakan peneliti dalam merancang sistem deteksi wajah, yaitu:

a) Python

Python adalah bahasa pemrograman tingkat tinggi yang bersifat serbaguna, mudah dipahami, dan digunakan secara luas dalam berbagai bidang pengembangan perangkat lunak. Dikembangkan pertama kali pada akhir 1980-an oleh Guido van Rossum, Python dirancang dengan fokus pada keterbacaan kode dan sintaks yang jelas, sehingga membuatnya menjadi bahasa yang cocok untuk pemula dan juga programmer berpengalaman.

b) OpenCV

OpenCV (Open-source Computer Vision Library) adalah perpustakaan perangkat lunak open-source yang dirancang untuk memungkinkan pengembang perangkat lunak mengembangkan aplikasi yang berhubungan dengan pengolahan citra dan analisis komputer vision. OpenCV menyediakan berbagai fungsi dan algoritma untuk memproses, menganalisis, dan memanipulasi data citra dan video.

c) Google Colaboratory

Google Colaboratory, atau yang biasa disingkat dengan Google Colab, adalah platform pengembangan berbasis cloud yang disediakan oleh Google untuk menjalankan kode Python secara interaktif. Ini dirancang khusus untuk keperluan pemrograman, analisis data, dan pembelajaran mesin. Google Colab memungkinkan pengguna untuk membuat dan menjalankan kode Python dalam lingkungan Jupyter Notebook yang dapat diakses melalui browser web.

2.2.8 White Box

Metode ini berfungsi untuk menguji sistematika aplikasi dengan memperhatikan struktur kode program yang sudah dirancang sebelumnya agar memaksimalkan skenario dalam pengujian aplikasi dan mengetahui kesalahan galat dalam implementasi kode program (Londjo, 2021). Pada dasarnya pengujian ini digunakan untuk menguji logika program dalam mendeteksi beberapa kondisi indikasi pelanggaran peserta asesmen ketika sudah diintegrasikan dengan model deteksi wajah menggunakan algoritma YOLOv8.

2.2.9 Black Box

Metode ini berfungsi untuk menguji sebuah aplikasi tanpa harus memperhatikan detail kode program dan mengetahui luaran setiap fungsi dapat berjalan dengan baik sesuai dengan rancangan/rencana yang sudah ditetapkan sebelumnya (Ningrum, Suherman, Aryanti, Prasetya, & Saifudin, 2019). Sedikit berbeda dengan pengujian *white box*, metode ini digunakan untuk menguji seluruh fungsionalitas sistem deteksi wajah menggunakan algoritma YOLOv8.