IMPLEMENTASI ALGORITMA YOLOv7 PADA SISTEM PENGHITUNG BENIH KELAPA SAWIT SECARA *REALTIME*

SKRIPSI

NASRULLAH M. HARIS

H071181320

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2023

IMPLEMENTASI ALGORITMA YOLOv7 PADA SISTEM PENGHITUNG BENIH KELAPA SAWIT SECARA *REALTIME*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

NASRULLAH M. HARIS

H071181320

PROGRAM STUDI SISTEM INFORMASI

DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2023

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: Nasrullah M. Haris
NIM	: H071181320
Program Studi	: Sistem Informasi
Jenjang	: Strata I (S1)

Menyatakan dengan ini bahwa karya tulisan saya yang berjudul

IMPLEMENTASI ALGORITMA YOLOv7 PADA SISTEM PENGHITUNG BENIH KELAPA SAWIT SECARA REALTIME

adalah karya tulisan saya sendiri dan bukan merupakan pengambilalihan tulisan orang lain dan belum pernah dipublikasikan dalam bentuk apa pun.

Apabila dikemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 12 Juli 2023

Yang menyatakan,

Nasrullah M. Haris
NIM: H071181320

IMPLEMENTASI ALGORITMA YOLOv7 PADA SISTEM PENGHITUNG BENIH KELAPA SAWIT SECARA *REALTIME*

Disusun dan dilakukan oleh

NASRULLAH M. HARIS

H071811320

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka penyelesaian studi Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui

Pembimbing Utama,	**Pembimbing Pertama,**
Dr.Eng. Armin Lawi, S.Si., M.Si	**Nanang Supena, M.P.**
NIP: 19720423 199512 1 001	NP: 200198105008

Ketua Program Studi,

Dr. Hendra, S.Si., M.Kom.
NIP: 19760102 200212 1 001

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama	: Nasrullah M. Haris
NIM	: H071181320
Program Studi	: Sistem Informasi
Judul Skripsi	: Implementasi Algoritma YOLOv7 Pada Sistem Penghitung Benih Kelapa Sawit Secara *Realtime*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan dan diperlukan untuk memperoleh gelar Sarjana Komputer pada Program Studi Sistem Informasi Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin.

DEWAN PENGUJI

		Tanda Tangan
Ketua	: Dr.Eng. Armin Lawi, S.Si., M.Eng.	(.....)
Sekretaris	: Nanang Supena, M.P.	(.....)
Anggota	: A. Muh. Amil Siddik, S.Si., M.Si	(.....)
Anggota	: Edy Saputra Rusdi, S.Si., M.Si	(.....)

Ditetapkan di : Makassar

Tanggal : 12 Juli 2023

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS HASANUDDIN DEPARTEMEN MATEMATIKA FAKULTAS MIPA

KATA PENGANTAR

Puji dan syukur kehadirat Allah SWT yang telah memberikan limpahan rahmat, dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul **"Implementasi Algoritma YOLOv7 Pada Sistem Penghitung Benih Kelapa Sawit Secara *Realtime*".**

Skripsi ini disusun untuk memenuhi sebagian persyaratan dalam menyelesaikan pendidikan pada jenjang Strata Satu (S1) pada Program Studi Sistem Informasi, Departemen Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Hasanuddin.

Penulis menyadari bahwa penyusunan skripsi ini tidaklah mudah, berbagai cobaan, kesulitan, dan hambatan yang penulis temui sejak dari awal pembuatan skripsi ini hingga menjelang penyelesaiannya. Namun dapat teratasi berkat tekad dan upaya keras serta tentunya dukungan dari berbagai pihak.

Salah satu kebanggaan yang akan selalu dikenang adalah ketika kita bisa melihat atau merasakan sebuah impian menjadi kenyataan. Bagi penulis, skripsi ini adalah salah satu impian yang diwujudkan dalam kenyataan dan dibuat dengan segenap kemampuan.

Pada kesempatan ini pula, penulis ingin mengucapkan rasa terima kasih kepada keluarga tercinta, kasih sayang yang tak terhingga dan penghormatan yang sebesar-besarnya penulis berikan kepada kedua orang tua penulis, yakni ibunda tercinta **Hj. Mardiati** dengan ayahanda **H. Muh. Haris**, atas segala perjuangan mendidik, membesarkan penulis sampai pada saat ini, memberikan dukungan serta doa yang tulus dan tak kenal lelah kepada penulis hingga dapat menyelesaikan studi, serta para saudara dan saudari saya, yang selalu mendorong saya agar dapat menyelesaikan studi dengan tepat waktu. Terima kasih atas nilai-nilai kehidupan yang senantiasa diberikan untuk terus menjadi pribadi yang lebih baik, pribadi yang terus berusaha melayakkan diri sebagai manusia yang dapat menjadi berkat bagi masyarakat, nusa dan bangsa. Semoga Allah SWT, memberikan kesehatan, dan melindungi setiap langkah kehidupan kepada keluarga penulis.

Terima kasih yang sebesar-besarnya serta penghargaan yang setinggi-tingginya juga penulis sampaikan kepada:

1. Bapak Prof. Dr. Ir. Jamaluddin Jompa, M.Sc. selaku Rektor Universitas Hasanuddin Makassar.
2. Bapak Dr.Eng. Amiruddin, S.Si., M.Si. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam serta para Wakil Dekan dan seluruh jajaran staf yang telah memberikan bantuan selama penulis mengikuti pendidikan.
3. Bapak Dr. Nurdin, S.Si., M.Si. selaku ketua Departemen Fakultas Matematika dan Ilmu Pengetahuan Alam. Beserta seluruh jajaran staf yang telah memberikan bantuan selama penulis mengikuti pendidikan.
4. Bapak Dr. Hendra, S.Si., M.Si. selaku Ketua Program Studi Sistem Informasi.
5. Bapak Dr.Eng. Armin Lawi, S.Si., M.Si. selaku dosen pembimbing utama dan Bapak Nanang Supena, M.P. selaku dosen pembimbing pertama untuk segala ilmu yang beliau berikan, kesabaran dalam membimbing, nasihat serta memotivasi penulis dalam penulisan skripsi ini.
6. Bapak A. Muh. Amil Siddik, S.Si., M.Si. dan Bapak Edy Saputra Rusdi, S.Si., M.Si. selaku dosen penguji yang telah bersedia meluangkan waktunya untuk memberikan saran dan arahan kepada penulis dalam penulisan skripsi ini.
7. Bapak/Ibu Dosen Pengajar Departemen Matematika, terkhusus kepada Bapak/Ibu Dosen Pengajar Program Studi Sistem Informasi yang telah memberikan ilmu kepada penulis selama menjadi mahasiswa di Program Studi Sistem Informasi.
8. Pusat Penelitian Kelapa Sawit dan Program Kampus Merdeka atas kesempatan yang diberikan untuk menimba ilmu dan memperoleh pengalaman yang luar biasa selama mengikuti Program Magang dan Studi Independen Bersertifikat Kampus Merdeka.
9. Bapak Muzakky Al Maududy, S.Si. selaku mentor selama mengikuti Program Magang dan Studi Independen Kampus Merdeka untuk segala ilmu yang beliau berikan, kesabaran dalam membimbing, serta memotivasi penulis dalam menyelesaikan skripsi ini.
10. Seluruh staf UKDP PPKS khususnya kepada bang Brian, bang Daus, dan Bu Melly yang telah banyak membantu selama mengikuti program MSIB di Pusat Penelitian Kelapa Sawit Kota Medan.

11. Tim AIWA (Ajie, Aldino, Ahlul, Fahroza) yang telah banyak membantu penulis dalam setiap suka dan duka penulis dalam menyelesaikan skripsi ini.
12. Seluruh teman-teman program MSIB Kampus Merdeka *batch* 3 di Pusat Penelitian Kelapa Sawit yang telah banyak membantu dan mendukung penulis dalam menyelesaikan skripsi ini.
13. Seluruh teman-teman Ilmu Komputer 2018 yang telah bersama-sama berjuang dalam setiap suka dan duka hingga dapat menyelesaikan pendidikan hingga di titik akhir.
14. Founder NAS ID, Nasmah Indah Sari yang telah banyak memberikan bantuan dan dukungan baik secara moril maupun materiil hingga dapat bersama-sama menyelesaikan studi dengan baik, terima kasih selalu ada.
15. Seluruh teman-teman Kuliah Kerja Nyata (KKN) G-106 Bulukumba 1, terima kasih untuk semua cerita dan pengalaman selama proses pelaksanaan KKN di Kabupaten Bulukumba. Semoga tetap diberi kesehatan dan waktu untuk mempertemukan.
16. Para Founder Yayasan Karya Salemba Empat beserta staf dan jajarannya serta para donatur Beasiswa KSE yang telah memberikan bantuan luar biasa serta pengalaman tak ternilai hingga penulis dapat menyelesaikan pendidikannya dengan baik.
17. Seluruh teman-teman Batch 4, adik-adik Beswan, dan kakanda Alumni Paguyuban KSE Unhas.

Untuk kesempurnaan skripsi ini, penulis mengharapkan kritik dan saran yang membangun dari semua pihak, semoga skripsi ini dapat bermanfaat untuk semua orang.

Akhirnya penulis mengucapkan terima kasih atas segala kebaikan dan bantuan yang di berikan semoga mendapat balasan yang setimpal dari Allah SWT.

Penulis,

Nasrullah M. Haris

PERNYATAAN PERSETUJUAN PUBLIKASI UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: Nasrullah M. Haris
NIM	: H071181320
Program Studi	: Sistem Informasi
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Eksklusif (*Non-exclusive Royalty-free Right*)** atas karya ilmiah saya yang berjudul:

Implementasi Algoritma YOLOv7 Pada Sistem Penghitung Benih Kelapa Sawit Secara *Realtime*

beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak universitas berhak menyimpan, mengalih-media/format-kan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenar-benarnya.
Dibuat di Makassar pada tanggal 12 Juli 2023

Yang menyatakan,

Nasrullah M. Haris

ABSTRAK

Penelitian ini bertujuan untuk mengembangkan sistem penghitung benih kelapa sawit secara *real-time* menggunakan algoritma YOLOv7. Dalam industri kelapa sawit di Indonesia, permintaan benih kelapa sawit unggul meningkat namun sering terjadi kesalahan penghitungan benih akibat penggunaan laser sebagai sensor. Penelitian ini mengusulkan penerapan YOLOv7 untuk mengatasi masalah tersebut.

Penelitian ini menggunakan 500 data gambar untuk melatih model YOLOv7 dengan *hyperparameter* yang ditentukan, dan memperoleh skor mAP sebesar 97,1%. Model tersebut diimplementasikan pada sistem penghitung benih kelapa sawit secara *real-time* dengan kecepatan deteksi 28,81 FPS pada kamera resolusi 720 piksel dan 24,02 FPS pada kamera resolusi 1080 piksel.

Implementasi sistem dimulai dengan membuat aplikasi *desktop* yang menggunakan model YOLOv7 untuk mendeteksi objek benih kelapa sawit pada video. Hasil penghitungan benih disimpan pada basis data lokal dan disinkronisasi dengan basis data *cloud* melalui aplikasi web. Data hasil penghitungan dikelola otomatis untuk menghasilkan laporan yang komprehensif.

Penelitian ini memberikan kontribusi dalam pengembangan teknologi untuk industri kelapa sawit dengan meningkatkan akurasi dan efisiensi penghitungan benih. Dengan penerapan YOLOv7, sistem penghitung benih kelapa sawit dapat bekerja secara *real-time* dan mengatasi masalah kesalahan penghitungan yang sering terjadi. Penelitian ini menunjukkan potensi penerapan kecerdasan buatan dalam industri kelapa sawit dan memberikan landasan untuk penelitian lebih lanjut dalam bidang ini.

Kata kunci: kecerdasan buatan, YOLOv7, deteksi objek, industri kelapa sawit

ABSTRACT

This research aims to develop a real-time palm oil seed counter system using the YOLOv7 algorithm. In the palm oil industry in Indonesia, there is an increasing demand for superior palm oil seeds, but there are often errors in seed counting due to the use of laser sensors. This study proposes the implementation of YOLOv7 to address this issue.

The research utilized 700 image data to train the YOLOv7 model with specified hyperparameters, achieving an mAP score of 97.1%. The model was implemented in a real-time palm oil seed counter system, achieving a detection speed of 28.81 FPS for a 720-pixel resolution camera and 24.02 FPS for a 1080-pixel resolution camera.

The system implementation began with the development of a desktop application that utilizes the YOLOv7 model to detect palm oil seed objects in videos. The seed counting results were stored in a local database and synchronized with a cloud-based database through a web application. The data were automatically managed to generate comprehensive reports.

This research contributes to the advancement of technology in the palm oil industry by enhancing the accuracy and efficiency of seed counting. The implementation of YOLOv7 enables the palm oil seed counting system to operate in real-time and overcome the frequent errors in counting. The study demonstrates the potential application of artificial intelligence in the palm oil industry and provides a foundation for further research in this field.

Keywords: artificial intelligence, YOLOv7, object detection, palm oil industry.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan salah satu negara produsen kelapa sawit terbesar di dunia dengan total produksi tahunan hingga 48,2 juta ton (Kementerian Pertanian, 2022). Industri kelapa sawit di Indonesia telah berkembang pesat sejak tahun 1970-an, dengan luas lahan tanaman kelapa sawit yang semakin meningkat dan produktivitas yang meningkat seiring dengan perkembangan teknologi dan budidaya. Perkembangan industri kelapa sawit di Indonesia juga didukung oleh pertumbuhan pasar dan permintaan akan produk-produk olahan kelapa sawit, seperti minyak kelapa sawit, margarin, biodiesel, dan lain-lain. Pemerintah Indonesia juga telah mengeluarkan berbagai kebijakan dan program untuk mendukung perkembangan industri kelapa sawit, seperti pemberian insentif fiskal dan dukungan teknis untuk pengembangan teknologi dan budidaya.

Perkembangan industri kelapa sawit tidak lepas dari kehadiran Pusat Penelitian Kelapa Sawit (PPKS) yang telah turut serta dalam mengembangkan inovasi yang mendukung kemajuan industri kelapa sawit. Salah satu inovasi yang telah dilakukan adalah pengembangan benih unggul kelapa sawit yang menjadi produk unggulan Pusat Penelitian Kelapa Sawit. Benih unggul yang dihasilkan oleh PPKS telah digunakan di banyak pertanian kelapa sawit di Indonesia. Berbagai keunggulan dari benih kelapa sawit PPKS menyebabkan tingginya permintaan benih dari petani baik individu maupun korporasi, sehingga banyak petani yang harus menunggu dalam waktu yang lama untuk mendapatkan benih unggul yang diinginkan.

Seiring dengan tingginya permintaan benih kelapa sawit unggul, PPKS juga turut meningkatkan kapasitas produksi benih untuk menutupi kebutuhan tersebut. Namun, pada proses produksi terdapat kendala dimana dalam proses penghitungan benih sering terjadi kesalahan yang menyebabkan adanya kesalahan penghitungan, hal ini tentu saja dapat merugikan baik petani maupun PPKS selaku produsen. Kesalahan tersebut terjadi dikarenakan alat penghitung benih yang saat ini digunakan masih menggunakan laser sebagai sensornya. Kekurangan laser dalam

alat penghitung tersebut adalah ketika dua buah benih bergerak beriringan, maka laser hanya mendeteksi satu buah benih saja. Hal tersebut dapat merugikan PPKS dikarenakan jumlah benih yang berlebih tidak masuk dalam jumlah yang dibayarkan oleh pembeli, dimana dalam beberapa kasus terkadang pembeli tidak menyampaikan perihal kelebihan jumlah benih yang diterima.

Salah satu teknologi yang dapat digunakan untuk mengurangi kesalahan tersebut adalah dengan menggunakan *artificial intelligence* (AI) atau kecerdasan buatan dalam bidang *computer vision* yang dapat menghitung benih secara akurat. Penerapan AI dalam industri kelapa sawit khususnya di PPKS sendiri masih terbilang langka, dikarenakan AI merupakan teknologi terbaru yang penerapannya masih minim, sehingga perlu lebih banyak penelitian untuk penerapan dalam skala besar.

Beberapa penelitian terkait *computer vision* khusus *object detection* telah banyak dilakukan, contohnya pada penelitian mengenai penggunaan algoritma YOLO untuk mendeteksi buah kelapa sawit dari kendaraan udara tanpa awak (UAV) yang dilakukan oleh Junos dkk., pada 2021. Pada penelitian tersebut, digunakan 700 data gambar yang dikumpulkan menggunakan kamera *mobile* dan kamera dari UAV. Adapun model yang digunakan adalah YOLOv3-*tiny* yang telah dioptimalkan untuk meningkatkan akurasi deteksi. Berdasarkan hasil pengujian model, diperoleh hasil presisi rata-rata 99,76% dengan skor F1 hingga 0,98 dan IoU rata-rata 83,26% dengan kecepatan deteksi hingga 34,06 milidetik (Junos dkk., 2022). Kemudian pada penelitian yang dilakukan oleh Parico dan Ahamed pada 2021 yang menggunakan YOLOv4 dan algoritma pelacakan multi object, DEEP SORT dengan menggunakan 448 data gambar buah pir yang dengan resolusi yang berbeda dan menghasilkan skor mAP hingga 96% dengan kecepatan deteksi hingga 37,3 FPS (Parico & Ahamed, 2021).

Berdasarkan uraian dan penelitian-penelitian di atas, telah banyak dilakukan penelitian terkait *object detection* untuk mendeteksi objek umum. Namun, belum banyak penelitian yang berkaitan dengan industri kelapa sawit khususnya di Pusat Penelitian Kelapa Sawit. Maka dari itu, penulis tertarik untuk melakukan penelitian dengan menggunakan algoritma YOLOv7 untuk menghitung benih kelapa sawit secara *real-time* dan akurat.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang, rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana melakukan *deployment* model hasil *training* dataset menggunakan YOLO-v7 untuk deteksi objek dan menghitung benih kelapa sawit secara *real-time*?
2. Bagaimana implementasi model hasil *training* dataset menggunakan YOLO-v7 pada aplikasi berbasis desktop dan menampilkan hasil penghitungan pada aplikasi berbasis web?

1.3 Batasan Masalah

Batasan masalah dalam penelitian ini sebagai berikut:

1. Data yang digunakan berupa data citra benih kelapa sawit dari Pusat Penelitian Kelapa Sawit unit Marihat. Data berupa gambar objek benih kelapa sawit yang ditebar di atas bidang datar dimana benih tidak saling menumpuk atau beririsan.
2. Luaran yang dihasilkan dari penelitian ini berupa aplikasi web dan *desktop* yang digunakan untuk penghitungan benih.
3. Aplikasi web dikembangkan menggunakan bahasa pemrograman PHP dengan *framework* CodeIgniter 3, sedangkan aplikasi *desktop* dikembangkan menggunakan bahasa pemrograman Python dengan GUI *PyQt5* serta menggunakan basis data MySQL.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah, berikut tujuan dari penelitian:

1. Model hasil *training* dataset menggunakan YOLO-v7 dilakukan *deployment* untuk deteksi objek dan menghitung benih kelapa sawit secara *real-time*.
2. Model hasil *training* dataset menggunakan YOLO-v7 diimplementasikan pada aplikasi penghitung benih kelapa sawit berbasis *desktop* dan menampilkan hasil penghitungan pada aplikasi web.

1.5 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Dapat mengembangkan metode penghitungan benih kelapa sawit yang lebih efektif dan akurat dengan mengimplementasikan algoritma YOLO dalam sistem penghitung benih kelapa sawit secara *real-time*.
2. Dapat meningkatkan efisiensi dalam proses penghitungan benih kelapa sawit, sehingga dapat membantu perusahaan dalam mengestimasi jumlah benih yang tersedia.
3. Dapat memberikan informasi yang berguna bagi pelaku industri tentang kemampuan algoritma YOLO, sehingga dapat membantu dalam pengambilan keputusan terkait perencanaan produksi dan distribusi suatu produk.
4. Dapat memberikan kontribusi bagi perkembangan ilmu pengetahuan dan teknologi dalam industri kelapa sawit menggunakan kecerdasan buatan.

BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Relevan

Penelitian yang dilakukan oleh Zaryanti Zainuddin pada 2022 mengenai sistem deteksi limbah pakan udang menggunakan algoritma YOLO menggunakan 370 data gambar. Dari pengujian yang telah dilakukan, sistem tersebut dapat menghasilkan nilai mAP hingga 96-97% pada *batch* maksimum 4000-10000 dan pada percobaan pada jarak 25 cm menghasilkan mAP terbaik hingga 83,31% (Zainuddin dkk., 2022).

Penelitian lainnya dilakukan oleh Zulkiflie pada 2021 mengenai sistem deteksi pelanggaran *social distancing* menggunakan YOLOv4 dan *Euclidean Distance* yang menggunakan 29.370 data citra kerumunan orang dimana pada penelitian tersebut digunakan YOLOv4 dengan *framework darknet*. Dari hasil pengujian deteksi manusia menggunakan video digital diperoleh skor mAP hingga 78.6% dengan kecepatan deteksi terbaik hingga 51,2 FPS (Zulkiflie, 2021).

Penelitian mengenai deteksi dan penghitungan buah pir pernah dilakukan oleh Parico pada 2021 menggunakan YOLOv4 dan algoritma pelacakan multi objek *DEEP SORT*. Pada penelitian tersebut digunakan 448 data gambar buah pir yang dengan resolusi yang berbeda dan menghasilkan skor mAP hingga 96% dengan kecepatan deteksi hingga 37,3 FPS (Parico & Ahamed, 2021).

Penelitian lainnya pernah dilakukan oleh Junos pada 2021 yang menggunakan algoritma YOLO untuk deteksi otomatis buah kelapa sawit dari kendaraan udara tanpa awak (UAV). Pada penelitian tersebut, digunakan 700 data gambar yang dikumpulkan menggunakan kamera *mobile* dan kamera dari UAV. Adapun model yang digunakan adalah YOLOv3-*tiny* yang telah dioptimalkan untuk meningkatkan akurasi deteksi. Berdasarkan hasil pengujian model, diperoleh hasil presisi rata-rata 99,76% dengan skor F1 hingga 0,98 dan IoU rata-rata 83,26% dengan kecepatan deteksi hingga 34,06 milidetik (Junos dkk., 2022).

Dalam penelitian yang berjudul "*Fruit and Vegetables Recognition using YOLO*" oleh Latha dkk. yang menggunakan algoritma YOLO dengan model YOLOv4-*tiny* untuk mendeteksi buah dan sayur secara *real-time*. Pada penelitian

tersebut digunakan 9800 data citra berbagai jenis buah dan sayur yang diperoleh dari sumber terbuka. Berdasarkan pengujian model pada sistem tertanam, diperoleh hasil deteksi dengan skor mAP 51% dengan kecepatan deteksi hingga 18 milidetik (Latha dkk., 2022).

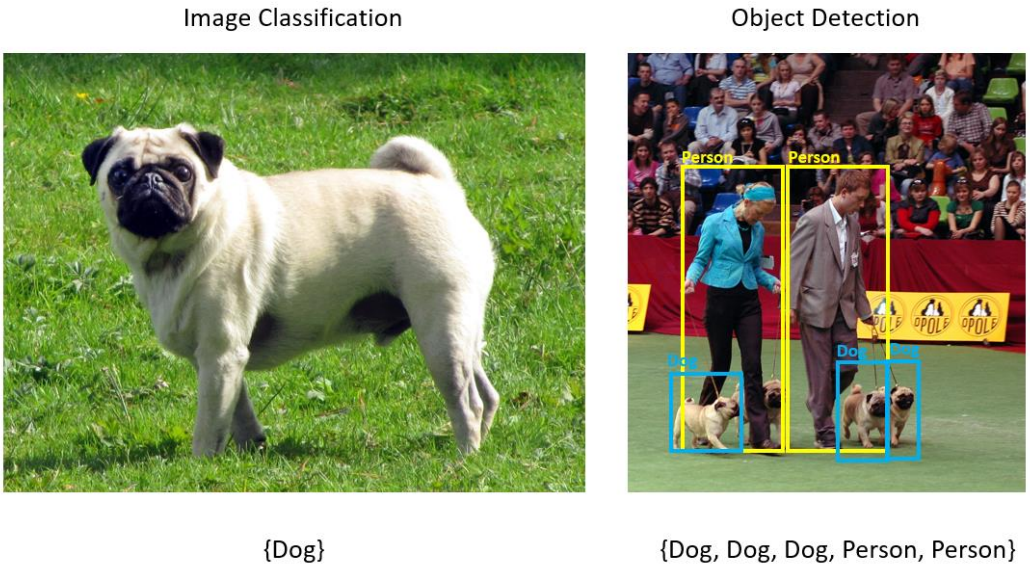
2.2 Kelapa Sawit

Kelapa sawit adalah salah satu jenis tanaman perkebunan yang dibudidayakan di Indonesia yang digunakan sebagai sumber minyak. Tanaman ini berasal dari hutan hujan tropis di Afrika Barat, terutama di Kamerun, Pantai Gading, dan Liberia. Kelapa sawit tidak merupakan tanaman asli dari Indonesia, Namun pertama kali ditemukan oleh Nicholas Jacquin pada tahun 1763. Pada tahun 1911, kelapa sawit mulai ditanam secara komersial di Indonesia (Lubis, 2008). Kelapa sawit pertama kali masuk ke Indonesia pada tahun 1848 melalui empat biji yang dibawa oleh Dr. D. T. Pryce. Tanaman ini awalnya ditanam sebagai tanaman hias di Kebun Raya Bogor (Supriyono, 2017), namun sejak saat itu kelapa sawit terus dikembangkan di berbagai daerah di Indonesia.

Terdapat tiga varietas dari kelapa sawit yang dibedakan berdasarkan ketebalan cangkangnya, yaitu dura, pisifera, dan tenera. Dura memiliki proporsi mesokarp (bagian yang mengandung minyak) dalam buah yang bervariasi antara 35-55%, meskipun ada yang mencapai 65%. Dura sangat cocok digunakan sebagai tanaman induk betina dalam produksi benih komersial. Selain itu, pisifera memiliki proporsi mesokarp dalam buah yang sangat tinggi dan rendemen minyak yang sangat tinggi (45-50%). Pisifera disebut juga sebagai pohon betina yang steril karena sebagian besar tandan yang dikembangkannya mengalami abortus pada awal perkembangannya. Oleh karena itu, digunakan sebagai tanaman induk jantan dalam produksi benih komersial. Pada benih unggul, digunakan tenera yang merupakan hasil persilangan antara dura dan pisifera. Tenera memiliki proporsi mesokarp dalam buah yang cukup tinggi mencapai (60-96%), yang merupakan tanaman kelapa sawit komersial yang ditanam untuk menghasilkan minyak kelapa sawit.

2.3 *Object Detection*

Deteksi objek atau *object detection* adalah salah satu bagian dari *computer vision* dan merupakan pengembangan dari klasifikasi gambar. Deteksi objek bertujuan untuk menemukan dan mengklasifikasikan objek yang ada dalam sebuah gambar atau video. Deteksi objek berbeda dengan klasifikasi, karena deteksi objek dapat mendeteksi lebih dari satu objek sekaligus serta memberikan informasi tentang lokasi objek dalam gambar. Detektor objek akan mengembalikan daftar objek yang terdeteksi dengan informasi kelas objek, probabilitas, dan koordinat objek untuk setiap objek yang terdeteksi (Vasilev dkk., 2019; Zhao dkk., 2018).

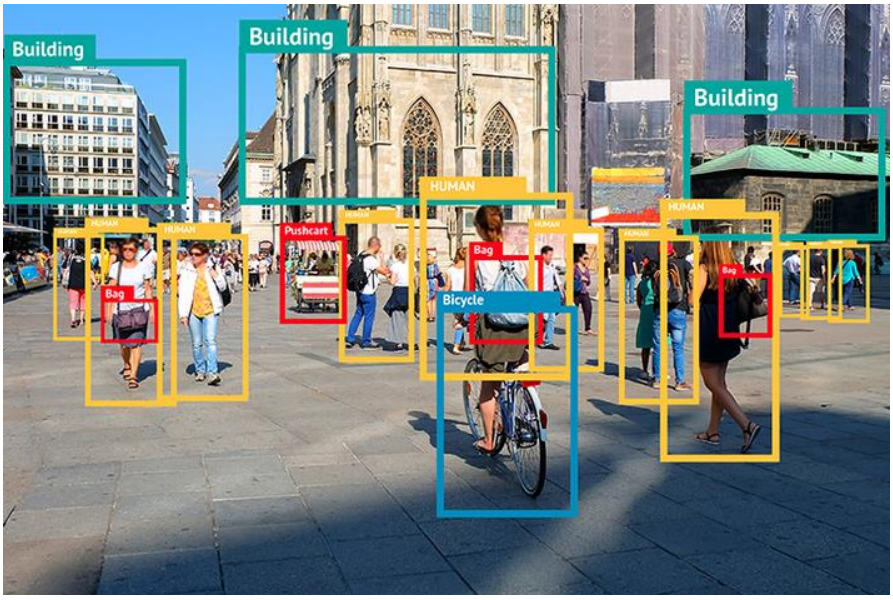


Gambar 2.1 Perbedaan Klasifikasi dan Deteksi Objek

(Sumber: *Detect stop signs in images with Model Builder* – learn.microsoft.com)

Pada **Gambar 2.1**, klasifikasi hanya dapat menentukan kelas dari sebuah objek pada gambar. Sedangkan, dalam deteksi objek terdapat banyak objek yang terdeteksi sekaligus dengan kelas yang beragam.

Dalam proses deteksi pada sebuah objek, detektor objek akan mengembalikan daftar objek yang terdeteksi berdasarkan informasi kelas objek (sendok, laptop, ponsel, benih) dan probabilitas atau skor keyakinan dalam rentang 0 dan 1 yang menunjukkan seberapa yakin detektor dalam menentukan keberadaan dan lokasi objek yang dideteksi. Selain itu, detektor juga mengembalikan informasi berupa koordinat wilayah dimana objek terdeteksi dalam bentuk persegi yang disebut sebagai *bounding box* sebagaimana yang ditunjukkan pada **Gambar 2.2**.



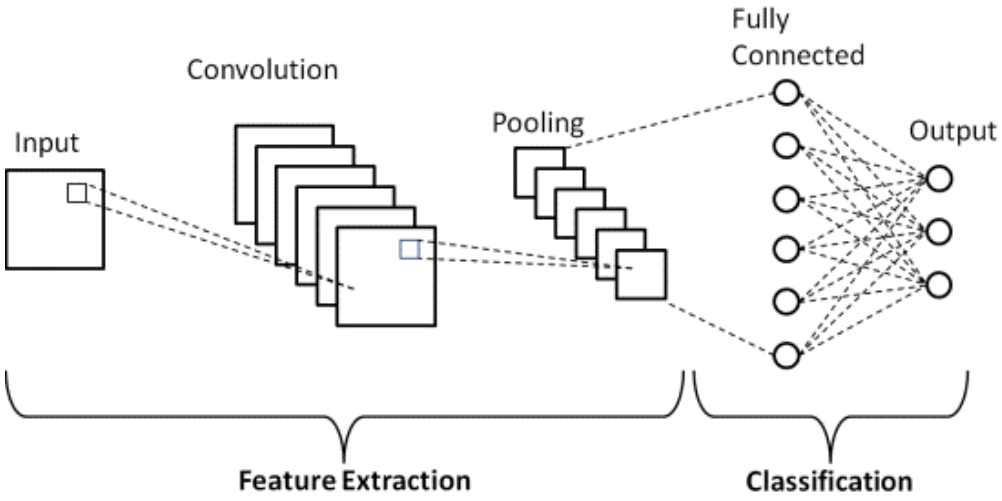
Gambar 2.2 *Bounding Box* pada Deteksi Objek

(Sumber: *Object Detection Technology – How It Works and Where Is It Used? –* DataScienceCentral.com).

2.4 *Convolutional Neural Network*

Convolutional Neural Network (CNN) adalah salah satu algoritma yang paling populer digunakan dalam *Deep Learning*. CNN merupakan metode yang diturunkan dari *Multi-Layer Perceptron* (MLP) untuk pengolahan data dalam bentuk *grid* dua dimensi. CNN digunakan untuk mengklasifikasikan data yang sudah diberi label dengan metode belajar berbasis supervisi. Algoritma ini sangat berguna untuk mencari pola dalam gambar dan mengenali objek di dalamnya.

Secara umum, CNN terdiri dari dua tahap, yaitu ekstraksi fitur atau *feature learning* dan klasifikasi dengan menggunakan *fully connected layer* (Ghosh dkk., 2020). Pada tahap ekstraksi fitur, gambar yang diinput akan diekstraksi untuk dipelajari nilainya. Nilai ini kemudian dikonversi menjadi vektor dan masuk ke tahap klasifikasi. Pada tahap klasifikasi, model jaringan saraf akan digunakan untuk mengklasifikasikan objek berdasarkan kelasnya. Model umum CNN terdiri dari empat komponen, yaitu lapisan konvolusi, lapisan *pooling*, fungsi aktivasi, dan lapisan *fully connected* (Indolia dkk., 2018).



Gambar 2.3 Arsitektur CNN

(Sumber: Katole dkk., 2015)

Pada **Gambar 2.3**, Proses ekstraksi fitur dalam CNN terdiri dari beberapa lapisan tersembunyi atau *hidden layer*, yaitu lapisan konvolusi dan *pooling*. CNN bekerja dalam hierarki, sehingga output dari lapisan konvolusi pertama digunakan sebagai input pada lapisan konvolusi berikutnya. Pada tahap klasifikasi, terdapat lapisan *fully connected* dan fungsi aktivasi yang menghasilkan output berupa hasil klasifikasi (Katole dkk., 2015).

2.4.1 *Convolution Layer*

Konvolusi adalah istilah matematis yang mengacu pada proses mengaplikasikan fungsi pada output dari fungsi lain secara berulang. Dalam pengolahan citra, konvolusi mengacu pada proses mengaplikasikan sebuah *kernel* atau matriks pada citra. *Kernel* adalah sebuah matriks kecil dengan tinggi dan lebarnya lebih kecil dari matriks citra yang akan di konvolusi. *Kernel* biasanya juga dikenal dengan istilah *filter* atau *convolution mask*. Tujuan dari melakukan konvolusi pada data citra adalah untuk mengekstrak fitur-fitur yang penting dari citra yang diinput. Dalam *machine learning*, input citra berupa *array* dua dimensi dan *kernel* adalah parameter yang berupa *array* multidimensi yang disesuaikan dengan model algoritma. Konvolusi dapat digunakan pada lebih dari satu dimensi. Sebagai contoh, jika menggunakan gambar dua dimensi sebagai input, maka *kernel* juga berbentuk dua dimensi sebagaimana dinyatakan pada persamaan (2.1).

$$S(i,j) = (I \times K)(i,j) = \sum_{a}\sum_{b} I(a,b)K(i-a,j-b) \tag{2.1}$$

dimana:

$S(i,j) =$ Fungsi hasil konvolusi

$I =$ Input

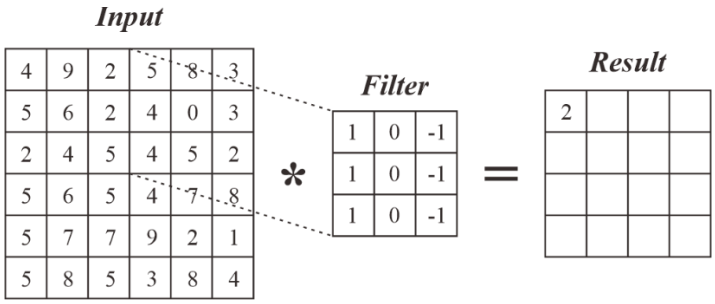
$K =$ *Kernel* atau *filter*

$(i,j) =$ *Pixel Kernel*

$(a,b) =$ *Pixel Kernel*

Lapisan konvolusi menggunakan filter untuk mengekstrak objek-objek dari citra yang diinput. Filter ini berisi bobot-bobot yang digunakan untuk mendeteksi karakteristik objek seperti tepi, kurva, atau warna. Konvolusi akan menghasilkan transformasi linear dari citra yang diinput yang sesuai dengan informasi spasial dari data. Filter diterapkan secara berulang sehingga menghasilkan serangkaian bidang reseptif. Ada parameter yang dapat diubah untuk memodifikasi sifat dari tiap lapisan, yaitu ukuran filter, *stride* dan *padding*.

Stride mengontrol bagaimana filter diterapkan pada data input dengan bergerak sepanjang jumlah piksel yang telah ditentukan. *Padding* adalah penambahan jumlah piksel dengan nilai tertentu di sekitar data input untuk menjaga agar hasil dari bidang reseptif tidak terlalu kecil sehingga tidak menyebabkan hilangnya banyak informasi. Nilai yang biasa digunakan adalah nol sehingga disebut dengan *zero padding*. Langkah ini melibatkan proses menggeser filter matriks (seperti ukuran 3x3) di atas gambar input yang memiliki ukuran lebar x tinggi. Filter pertama ditempatkan di atas matriks gambar, lalu menghitung perkalian elemen antara filter dan bagian gambar yang tumpang tindih, diikuti dengan penjumlahan untuk memberikan nilai fitur (Bahuleyan, 2018).



Gambar 2.4 Lapisan Konvolusi dengan *Filter* 3×3 dan *Stride* 1

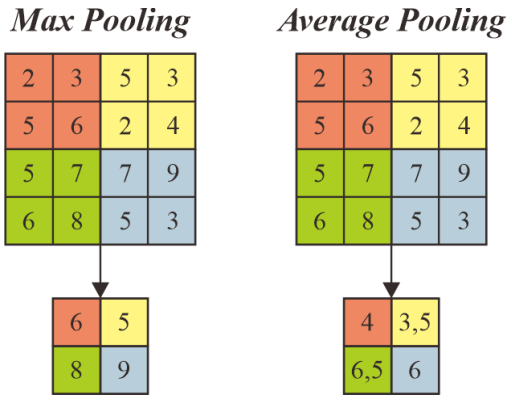
Seperti pada **Gambar 2.4** nantinya citra input akan terbentuk sebuah matriks 3×3 yang bergeser sesuai dengan *stride* yang sudah ditentukan. Kemudian matriks yang terbentuk akan dilakukan kuantisasi pada *filter*. Hasil dari kuantisasi

matriks berupa satu nilai output. Nantinya output kuantisasi dari tiap matriks pada citra akan membentuk sebuah matriks yang merupakan hasil dari konvolusi.

2.4.2 *Pooling Layer*

Lapisan *pooling* bertujuan untuk mengurangi ukuran dimensi dari data, sehingga mempermudah model untuk mempelajari pola-pola yang terdapat dalam data tersebut. Ada dua jenis lapisan pooling yang sering digunakan yaitu *max pooling* dan *average pooling*. *Max pooling* akan mengambil nilai tertinggi dari sekelompok data, sedangkan *average pooling* akan menghitung rata-rata dari sekelompok data. Dengan *pooling*, model CNN dapat mempelajari fitur yang tidak terpengaruh oleh perubahan skala dan mengurangi kompleksitas komputasi (Nirthika dkk., 2022).

Langkah melakukan pooling adalah dengan membagi hasil keluaran lapisan konvolusi menjadi beberapa grid berdasarkan jumlah *stride* dan jenis *pooling* yang digunakan. Terdapat dua jenis pooling yang umum digunakan pada lapisan *pooling*, yaitu *max pooling* dan *average pooling*. Pada *max pooling*, nilai yang diambil adalah nilai tertinggi dari setiap *grid*. Sedangkan pada *average pooling*, nilai yang diambil adalah nilai rata-rata dari setiap *grid*. Perbedaan dari *max pooling* dan *average pooling* bisa dilihat pada **Gambar 2.5**.



Gambar 2.5 *Max Pooling* dan *Average Pooling*

Pada proses *pooling*, hasil dari proses konvolusi akan terbentuk sebuah matriks 2×2, nantinya tiap matriks tersebut akan diubah menjadi satu elemen matriks saja.

2.4.3 *Activation Function*

Fungsi aktivasi bertugas untuk mengubah output dari sebuah layer menjadi nilai yang dapat diterima oleh layer. Fungsi aktivasi ini dapat mengubah output dari

layer menjadi nilai yang lebih terbatas, sehingga membantu model untuk mempelajari pola-pola yang ada dalam data dengan lebih baik (Sitepu & Sigiro, 2021).

Sigmoid adalah fungsi aktivasi nonlinear yang umum dimana output dari fungsi ini dibatasi dan fungsi ini sudah banyak digunakan sebagai fungsi aktivasi dalam *deep neural network* (Nwankpa dkk., 2018). Fungsi aktivasi *sigmoid* memiliki bentuk seperti huruf S, dan dapat membantu model untuk mempelajari pola-pola yang terdapat dalam data. Fungsi *sigmoid* akan mengubah output dari *layer* menjadi nilai yang berada di antara 0 dan 1. Hal ini berguna untuk mengatur *range* dari output *layer*, sehingga memudahkan model untuk mempelajari pola-pola yang ada dalam data. Fungsi aktivasi *sigmoid* baik digunakan pada kasus klasifikasi multi label, yaitu mengklasifikasi sesuatu yang memiliki jawaban lebih dari satu. Fungsi aktivasi *sigmoid* didefinisikan pada persamaan (2.2).

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

dimana

$x =$ data input; dan

$e \approx 2{,}718281828459045\ldots$

Fungsi aktivasi *sigmoid* dengan *threshold* berada di 0,5 yang hanya dapat memutuskan bahwa input yang diberikan hanya memiliki dua tipe kelas.

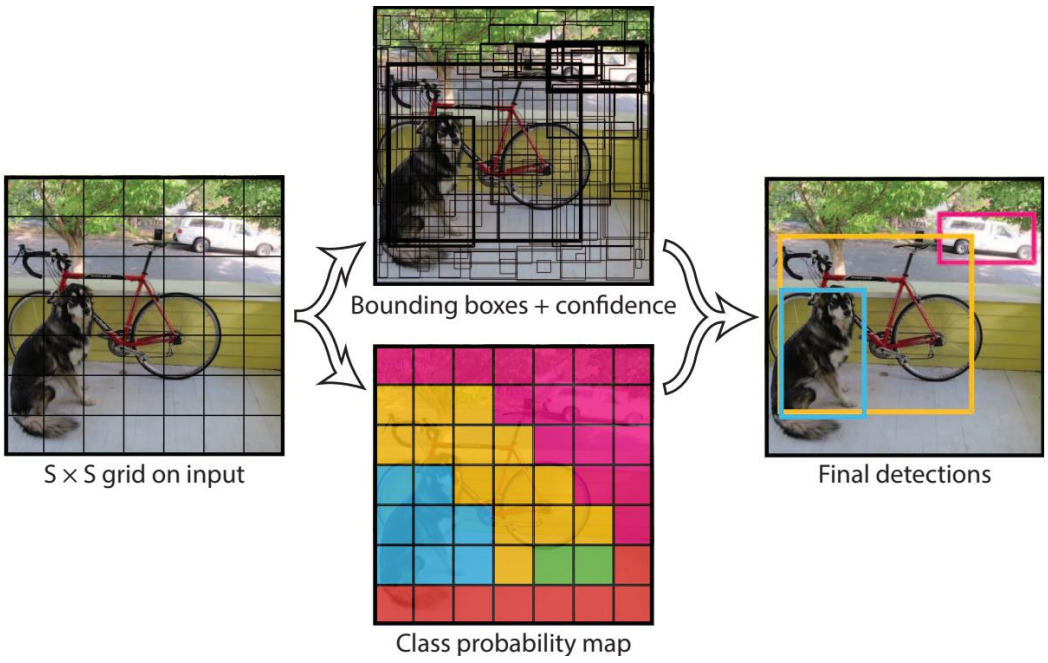
2.4.4 *Fully Connected Layer*

Lapisan *fully connected* merupakan lapisan yang bertujuan untuk menggabungkan fitur yang dipelajari dari lapisan sebelumnya dan menggunakannya untuk membuat prediksi atau keputusan. Lapisan ini disebut "*fully connected*" karena setiap neuron pada lapisan tersebut terhubung dengan setiap neuron pada lapisan sebelumnya. Data dari setiap neuron pada lapisan konvolusi harus ditransformasikan menjadi data satu dimensi terlebih dahulu sebelum dimasukkan ke dalam lapisan *fully connected*. Karena proses ini menyebabkan hilangnya informasi spasial dan tidak dapat dikembalikan, lapisan *fully connected* hanya dapat diimplementasikan di akhir jaringan (Ilahiyah & Nilogiri, 2018).

2.5 *You Only Look Once* (YOLO)

YOLO atau *You Only Look Once* merupakan salah satu algoritma deteksi objek yang populer. YOLO populer karena mencapai akurasi tinggi sekaligus dapat berjalan secara real-time. Algoritma "*you only look once*" pada gambar dalam arti hanya membutuhkan satu perambatan maju melewati jaringan saraf tunggal untuk melakukan klasifikasi dan prediksi kotak pembatas untuk objek yang terdeteksi. Dengan demikian, kinerja deteksi sangat dioptimalkan dan dapat berjalan lebih cepat daripada menjalankan dua jaringan saraf terpisah untuk mendeteksi dan mengklasifikasikan objek secara terpisah.

YOLO menggunakan pendekatan yang berbeda, YOLO adalah jaringan saraf konvolusi (CNN) yang dapat melakukan deteksi objek secara *real-time*. Algoritma ini menerapkan jaringan saraf tunggal pada gambar penuh, lalu membagi gambar menjadi beberapa wilayah dan memprediksi kotak batas dan probabilitas untuk setiap wilayah. Kotak batas ini diberi bobot berdasarkan probabilitas yang diprediksi.

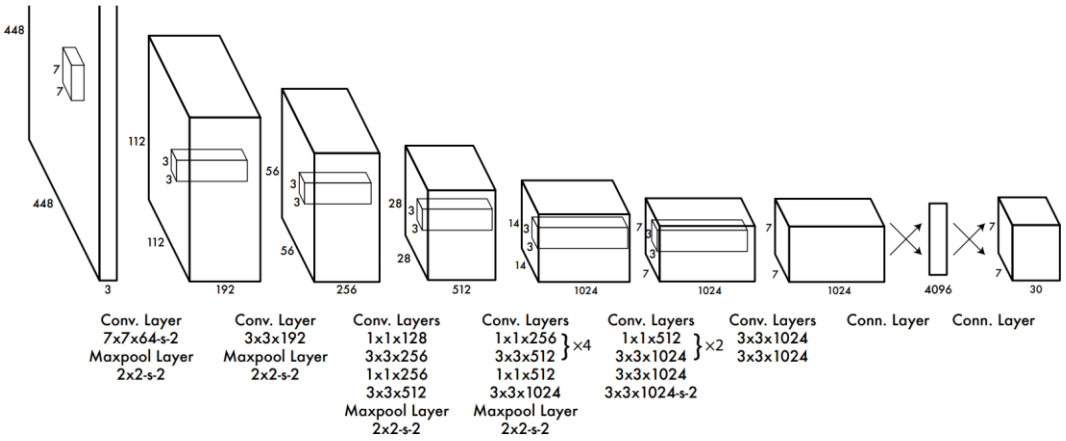


Gambar 2.6 Ilustrasi Cara Kerja YOLO

(Sumber: Redmon dkk., 2016)

Pada **Gambar 2.6** citra dibagi menjadi *grid* atau kisi dengan dimensi S×S dan untuk setiap sel pada kisi memprediksi B *bounding box*, nilai *confidence* untuk

setiap kotak, dan nilai probabilitas kelas C. Prediksi ini dikodekan sebagai tensor S × S × (B ∗ 5 + C) (Redmon dkk., 2016). Secara umum, arsitektur YOLO digambarkan sebagai *convolution layer* yang berlapis-lapis.



Gambar 2.7 Arsitektur YOLO

(Sumber: Redmon dkk., 2016)

Pada **Gambar 2.7** jaringan deteksi pada YOLO memiliki 24 lapisan konvolusi diikuti oleh 2 lapisan yang terhubung sepenuhnya. Lapisan *convolutional* 1×1 bergantian mengurangi ruang fitur dari lapisan sebelumnya. Lapisan konvolusional dilatih menggunakan klasifikasi ImageNet dengan setengah resolusi (gambar masukan 224×224) dan kemudian menggandakan resolusi untuk deteksi menjadi 448×448 (Redmon dkk., 2016).

Dalam perjalanannya, YOLO telah berkembang dengan performa lebih baik dibanding YOLO versi terdahulunya, terbaru adalah YOLOv7 (Wang dkk., 2022) yang menggunakan YOLOv4 dan YOLOR sebagai *baseline*.

Tabel 2.1 Perbandingan Model YOLO

(Sumber: Redmon dkk., 2016 - telah diolah kembali)

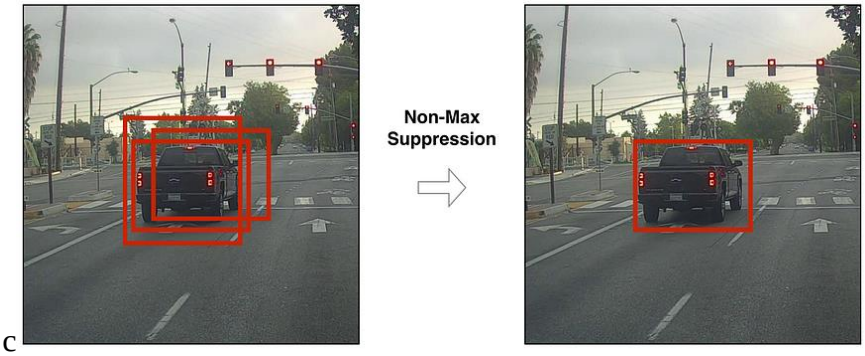
Model	Parameter	FLOPs	Ukuran	AP
YOLOv4	64,4M	142,8G	640	49,70%
YOLOR	52,9M	120,4G	640	50,80%
YOLOv7	36,9M	104,7G	640	51,20%

Pada **Tabel 2.1** YOLOv7 menggunakan parameter 43% lebih sedikit (36,9 *Megabyte*), komputasi 27% lebih sedikit (104,7 *Giga FLOP*), dan presisi rata-rata (AP) lebih tinggi 3% dibanding YOLOv4. Kemudian jika dibandingkan dengan

YOLOR, YOLOv7 menggunakan parameter 30% lebih sedikit, komputasi 13% lebih sedikit, dan presisi rata-rata lebih tinggi 0,78%. Kedua perbandingan tersebut merupakan hasil latihan model menggunakan dataset dengan ukuran citra yang sama, yaitu Ms COCO Dataset dengan ukuran 640×640 *pixel* (Wang dkk., 2022).

2.6 *Non-maximum Suppression* (NMS)

Non-maximum suppression (NMS) adalah algoritma *post-processing* yang digunakan untuk mengurangi *false positive* dan meningkatkan akurasi dalam *object detection* (Song dkk., 2019) seperti pada **Gambar 2.8**.



c

Gambar 2.8 Perbedaan Sebelum dan Setelah Menggunakan Algoritma NMS

(Sumber: *Non-maximum Suppression (NMS) - A technique to filter the predictions of object detectors* - towardsdatascience.com).

Algoritma ini dimulai dengan daftar kotak deteksi B dengan skor S. NMS memilih kotak deteksi dengan skor tertinggi M, menghapusnya dari daftar B, dan menambahkannya ke daftar deteksi akhir D. NMS juga menghapus kotak deteksi mana pun yang memiliki tumpang tindih yang lebih besar dari ambang batas Nt dengan M di daftar B. Proses ini diulang untuk kotak yang tersisa di B hingga B kosong. Algoritma NMS dapat dituliskan dengan *pseudocode* pada **Gambar 2.9**

```
Input: B = {b1, ..., bN}, S = {s1, ..., sN}, Nt
    B adalah daftar kotak deteksi awal
    S berisi skor deteksi yang sesuai
    Nt adalah ambang batas NMS

Output: D, S
    D adalah daftar kotak deteksi hasil akhir
    S berisi skor deteksi yang sesuai dengan deteksi pada D

BEGIN
    D = {} # inisialisasi daftar deteksi akhir
    while B is not empty do
        # Cari kotak deteksi dengan skor tertinggi
```

```
        m = argmax(S)
        M = B[m]
          # Tambahkan kotak deteksi dengan skor tertinggi ke daftar
deteksi akhir
        D.append(M)
         # Hapus kotak deteksi dengan skor tertinggi dari daftar kotak
deteksi dan skor deteksi
        B.remove(M)
        S.remove(S[m])
        # Hapus kotak deteksi dengan tumpang tindih yang signifikan
        for bi in B:
            if IoU(M, bi) > Nt:
                B.remove(bi)
                S.remove(S[B.index(bi)])
        end
    end
return D, S # Kembalikan daftar deteksi akhir dan skor deteksi yang
sesuai
END
```

Gambar 2.9 *Pseudocode* Algoritma NMS

Algoritma *non-maximum suppression* menetapkan ambang batas IoU (*Intersection over Union*) untuk objek kategori tertentu. *Bounding box* M dengan skor tertinggi dipilih dari serangkaian *bounding box* yang dihasilkan B. Kotak M dihapus dari B dan ditempatkan di hasil deteksi akhir R. Pada saat yang sama, *bounding box* dengan IoU dari M lebih besar dari ambang batas dihapus dari B (Bodla dkk., 2017).

Algoritma *non-maximum suppression* didefinisikan menggunakan persamaan berikut:

$$s_i = \begin{cases} s_{i,} & IoU(M, B_i) < p \\ 0, & IoU(M, B_i) \geq p \end{cases} \tag{2.3}$$

dimana s_i merupakan skor dari *bounding box* i. $IoU(M, B_i)$ merupakan nilai IoU antara *bounding box M* (*bounding box* dengan skor tertinggi) dengan *bounding box* i. p merupakan *threshold* nilai IoU yang ditentukan (Cao sdkk., 2019).

Jika nilai IoU antara *bounding box M* dengan *bounding box* i lebih kecil dari p, maka skor *bounding box* i tetap sama (tidak ditetapkan menjadi 0) karena masih dianggap sebagai objek yang berbeda. Jika nilai IoU antara *bounding box M* dengan *bounding box* i lebih besar dari atau sama dengan p, maka skor *bounding box* i akan ditetapkan menjadi 0. Hal ini dilakukan untuk menghindari adanya deteksi duplikat dari objek yang sama.

Penggunaan algoritma NMS dalam *object detection* juga memiliki kekurangan dimana skor untuk *bounding box* tetangga yang terdeteksi ditetapkan menjadi nol, yang dapat menyebabkan beberapa objek yang tumpang tindih terlewatkan. Untuk mengatasi masalah ini, dapat digunakan algoritma NMS yang dimodifikasi, yang menurunkan skor deteksi sebagai fungsi tumpang tindihnya dengan M. Dengan cara ini, objek tetap ada dalam daftar peringkat, meskipun dengan kepercayaan yang lebih rendah (Cao dkk., 2019).

2.7 *Hyperparameter*

Hyperparameter adalah variabel yang nilainya mengontrol proses *training* dan menentukan nilai parameter model. *Hyperparameter* merupakan salah satu faktor yang mempengaruhi kinerja dari model yang dibuat untuk mendapatkan performa yang baik. *Hyperparameter* tidak bisa didapatkan secara langsung pada proses *training*, melainkan dari inisiasi secara manual sebelum *training* dilakukan. Pada YOLO terdapat 31 *hyperparameter*, namun yang umum digunakan yaitu *optimizer*, *learning rate*, *epoch*, *batch*, dan *weight*.

2.7.1 *Optimizer* dan *Learning Rate*

Optimizer adalah algoritma yang digunakan untuk mengubah atribut jaringan saraf seperti *weight* dan *learning rate* untuk meminimalkan *cost function*, YOLOv7 menggunakan *Stochastic Gradient Descent* (SGD) dan Adam sebagai *optimizer*.

SGD adalah algoritma *optimizer* yang memperkirakan gradien kesalahan untuk status model saat ini menggunakan contoh dari kumpulan data pelatihan, lalu memperbarui bobot model menggunakan *backpropagation* dari kesalahan algoritma, yang disebut sebagai propagasi balik sederhana. Jumlah bobot yang diperbarui selama pelatihan disebut sebagai ukuran langkah atau "*learning rate*". Secara khusus, *learning rate* adalah *hyperparameter* yang dapat dikonfigurasi dan digunakan dalam pelatihan jaringan saraf yang memiliki nilai positif kecil, seringkali berkisar antara 0,0 dan 1,0 (Doshi, 2019).

Learning rate adalah *hyperparameter* yang mengontrol seberapa cepat model disesuaikan dengan masalah. Nilai yang lebih kecil memerlukan waktu pelatihan yang lebih lama karena perubahan yang lebih kecil pada bobot setiap kali

diperbarui, sementara nilai yang lebih besar akan menghasilkan perubahan cepat dan memerlukan waktu pelatihan yang lebih singkat. Namun, jika *learning rate* terlalu besar, model dapat konvergen ke solusi yang kurang optimal dan jika terlalu kecil, proses pelatihan dapat menjadi lambat dan tidak stabil (Doshi, 2019).

2.7.2 *Epoch*

Epoch adalah jumlah iterasi pelatihan yang digunakan untuk mengolah semua *batch* data masuk ke dalam *optimizer* dan keluar dari *optimizer*. Dalam satu *epoch*, seluruh dataset melalui proses pelatihan dan diulang kembali dari awal. Nilai *epoch* ditentukan secara eksperimental dan dapat dimulai dari angka yang kecil seperti 10 dan ditingkatkan sampai loss function tidak mengalami penurunan yang signifikan. Semakin banyak *epoch* yang digunakan, waktu pelatihan juga akan bertambah (Emanuella dkk., 2022).

2.7.3 *Batch Size*

Salah satu *hyperparameter* utama yang perlu disesuaikan sebelum memulai proses pelatihan adalah *batch size*, yaitu jumlah gambar yang digunakan di setiap *epoch* untuk melatih jaringan. Konfigurasi *hyperparameter* yang terlalu tinggi dapat membuat jaringan membutuhkan waktu terlalu lama untuk mencapai konvergensi (tidak ada lagi peningkatan akurasi), namun jika terlalu rendah akan membuat jaringan bolak-balik tanpa mencapai kinerja yang dapat diterima. Selain itu, sifat dari *dataset* dapat berdampak pada ukuran *batch* (Kandel & Castelli, 2020).

2.7.4 *Weight*

Weight atau bobot adalah koefisien dari fungsi yang akan diselesaikan. Bobot adalah nilai nyata yang diasosiasikan dengan setiap fitur yang menunjukkan pentingnya fitur tersebut dalam memprediksi nilai akhir. Ketika *neural network* dilatih pada *dataset*, maka akan terjadi inisialisasi sejumlah bobot. Bobot ini akan terus dioptimasi selama masa pelatihan dan akan mengeluarkan bobot paling optimum sebagai hasil.

2.8 Evaluasi Kinerja Model

Dalam pengembangan sebuah model deteksi objek, evaluasi kinerja model adalah hal penting yang perlu dilakukan. Hal ini bertujuan untuk mengukur kinerja

suatu model pada objek yang digunakan, adapun metode evaluasi yang umum digunakan adalah *Intersection over Union* (IoU), *precision* & *recall*, *Average Precision*, dan *mean Average Precision* (mAP).

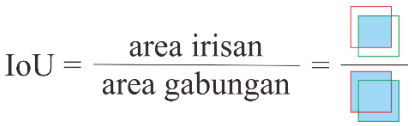
2.8.1 *Intersection over Union* (IoU)

Intersection over Union (IoU) adalah metrik yang digunakan untuk mengevaluasi seberapa baik hasil prediksi *bounding box* prediksi dengan *bounding box ground truth* (kebenaran). IoU dihitung dengan mengukur luas tumpang tindih (luas area yang beririsan) antara dua *bounding box* dan membandingkannya dengan luas total dari kedua *bounding box*. Nilai IoU menunjukkan seberapa baik kedua *bounding box* saling beririsan dan digunakan untuk mengevaluasi performa model deteksi objek.

Dengan menerapkan IoU kita dapat mengetahui apakah suatu deteksi valid (*True Positive*) atau tidak (*False Positive*). IoU dihasilkan oleh area yang tumpang tindih antara *bounding box* yang diprediksi dan *bounding box* kebenaran dasar (*ground truth*) dibagi dengan area gabungan di antara keduanya (Padilla dkk., 2021). IoU dapat dinyatakan pada persamaan (2.4).

$$IoU = \frac{B_{prediksi} \cap B_{groundTruth}}{B_{prediksi} \cup B_{groundTruth}} \tag{2.4}$$

dan dapat diilustrasikan pada gambar berikut:



Gambar 2.10 Ilustrasi Persamaan IoU

(Sumber: Padilla dkk., 2021 – telah diolah kembali)

Pada **Gambar 2.10** dapat dilihat bahwa persamaan untuk mendapatkan nilai IoU hanyalah sebuah perbandingan dari area irisan dibagi dengan area gabungan. Ketentuan untuk menilai apakah skor IoU yang diperoleh baik atau buruk adalah semakin beririsan atau semakin dekat jarak antara *bounding box* prediksi dengan *bounding box ground truth* (area yang beririsan antara kedua *bounding box* semakin besar), maka skor akan semakin tinggi.

2.8.2 *Precision-Recall*

Precision adalah kemampuan model untuk mengidentifikasi objek yang relevan saja sebagai persentase prediksi positif yang benar. Sedangkan *recall* adalah

adalah kemampuan model untuk menemukan semua kasus yang relevan (semua *bounding box* dari *ground truth*) sebagai persentase prediksi positif yang benar di antara semua *ground truth* yang diberikan. Penilaian metode deteksi objek sebagian besar didasarkan pada *precision* dan *recall* yang dinyatakan pada persamaan (2.5) dan (2.6).

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{semua\ deteksi} \tag{2.5}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{semua\ ground\ truth} \tag{2.6}$$

dimana *True Positive* (TP) adalah deteksi yang benar dari *bounding box ground truth*, *False Positive* (FP) adalah deteksi yang salah dari objek yang tidak ada atau deteksi yang salah dari objek yang ada, dan *False Negative* (FN) adalah *bounding box* dari *ground truth* yang tidak terdeteksi (Padilla dkk., 2020).

Jika jumlah *False Positive* rendah, maka nilai precision akan tinggi tetapi jumlah objek yang terlewatkan juga akan tinggi, sehingga menyebabkan nilai *False Negative* tinggi dan *recall* rendah. Namun, jika objek diterima dengan menurunkan ambang batas IoU, maka recall akan meningkat tetapi jumlah False Positive juga akan meningkat, yang akan menurunkan nilai precision. Untuk model yang baik, kedua nilai precision dan recall harus tetap tinggi meskipun ambang batas IoU bervariasi.

2.8.3 *mean Average Precision* (mAP)

mean Average Precision atau mAP adalah metrik yang digunakan untuk mengukur keakuratan detektor objek di semua kelas dalam *dataset* tertentu. Sebelum menentukan nilai mAP, perlu diketahui lebih dahulu nilai dari *Average Precision* (AP) yang diperoleh dengan menghitung *Area Under Curve* (AUC) dari kurva *precision-recall*. Namun, kurva *precision-recall* seringkali berbentuk zig-zag yang naik dan turun yang menyebabkan sulitnya menentukan estimasi nilai AUC. Untuk mengatasi hal tersebut digunakan dua pendekatan, yaitu interpolasi 11 titik dan interpolasi semua titik.

Pada interpolasi 11 titik, interpolasi dilakukan pada nilai *precision* yang terdekat pada 11 titik *recall* yang terdiri dari [0.0, 0.1, 0.2, …, 0.9, 1.0], interpolasi 11 titik dinyatakan pada persamaan (2.7).

$$AP_{11} = \frac{1}{11}\sum_{R\in\{0,0.1,\ldots,0.9,1.0\}} P_{interp}R, \quad (2.7)$$

dimana

$P_{interp}R = \max_{\tilde{R}:\tilde{R}\geq R} P(\tilde{R})$.

Artinya alih-alih mengambil nilai *precision* sebenarnya pada titik $\tilde{R}$, nilai *precision* tertinggi diambil dari yang terdekat dengan $\tilde{R}$ tetapi hanya mengambil titik setelah $\tilde{R}$ saja.

Pada interpolasi semua titik, alih-alih menginterpolasi hanya 11 titik dengan jarak yang sama, interpolasi dapat dilakukan melalui semua titik sedemikian rupa sehingga AP untuk semua titik dapat dinyatakan pada persamaan (2.8).

$$AP_{semua} = \sum_n (R_{n+1} - R_n) P_{interp}(R_{n+1}), \quad (2.8)$$

dimana

$P_{interp}(R_{n+1}) = \max_{\tilde{R}:\tilde{R}\geq R_{n+1}} P(\tilde{R})$.

Dalam hal ini, alih-alih menggunakan presisi yang diamati hanya pada beberapa titik, AP kini diperoleh dengan menginterpolasi presisi pada setiap level, mengambil presisi maksimum yang memiliki nilai *recall* lebih besar atau sama dengan R_{n+1}.

Pada dasarnya, *mean Average Precision* (mAP) hanyalah AP rata-rata dari semua kelas yang dinyatakan pada persamaan berikut:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (2.9)$$

dengan AP_i adalah kelas ke i dan N adalah total kelas yang dievaluasi (Padilla dkk., 2020).

2.9 *Pytorch* dan *Torchvision*

PyTorch merupakan pustaka sumber terbuka untuk pembelajaran mesin yang pertama kali dikembangkan oleh Facebook Research. *PyTorch* digunakan sebagai pengganti *numpy* untuk menggunakan kekuatan GPU dalam proses pembelajaran mesin. Selain itu, *PyTorch* juga digunakan sebagai *platform* penelitian pembelajaran mesin yang memberikan fleksibilitas dan kecepatan maksimum. *PyTorch* menyediakan Tensor yang dapat hidup di CPU atau GPU dan mempercepat komputasi dalam jumlah besar.

Salah satu komponen utama pada *PyTorch* adalah *Torch* yang berisi struktur data untuk tensor multidimensi dan menentukan operasi matematika pada tensor layaknya *numpy*. Pada pengembangan sistem deteksi objek menggunakan *PyTorch*, digunakan *Torchvision* yang merupakan pustaka *computer vision* yang bergandengan dengan *PyTorch*. *Torchvision* memiliki utilitas untuk transformasi gambar dan video yang efisien, beberapa model terlatih yang umum digunakan, dan beberapa kumpulan data yang umum (Nielsen, 2021).

2.10 *CodeIgniter*

CodeIgniter adalah sebuah framework PHP *open-source* yang digunakan untuk membangun aplikasi web dinamis. Hal ini mempercepat proses pengembangan aplikasi web dengan menyediakan dokumentasi lengkap yang disertai dengan contoh kode implementasi. *CodeIgniter* menggunakan konsep dasar *Model-View-Controller* (MVC) yang membagi aplikasi perangkat lunak menjadi tiga komponen utama: model, *view*, dan *controller*.

1. Model merupakan bagian dari aplikasi yang menangani aspek logika bisnis dan mengakses data. Ini menangani pertanyaan seperti "apa yang harus dilakukan" dan "bagaimana caranya".
2. *View* merupakan bagian dari aplikasi yang menangani aspek tampilan *user interface* (UI), seperti halaman web atau formulir yang ditampilkan kepada pengguna. Ini menangani pertanyaan seperti "apa yang harus ditampilkan" dan "bagaimana caranya".
3. *Controller* merupakan bagian dari aplikasi yang menangani aspek pengendalian, seperti menangani permintaan pengguna dan mengatur interaksi antara model dan *view*. Ini menangani pertanyaan seperti "apa yang harus dilakukan saat pengguna mengklik tombol" dan "bagaimana caranya".

Dengan memisahkan aplikasi ke dalam tiga komponen utama ini, MVC memungkinkan pengembang untuk lebih mudah mengelola dan memperbaiki aplikasi, serta memudahkan pemeliharaan dan pengembangan lebih lanjut.

2.11 Web API

API atau *Application Programming Interface*, adalah sekumpulan perintah, protokol, dan tools untuk membangun aplikasi perangkat lunak. API memungkinkan aplikasi untuk berkomunikasi satu sama lain dan berbagi data dan fungsi-fungsi yang dapat digunakan oleh aplikasi lain.

API yang dibangun menggunakan teknologi web disebut sebagai Web API, misalnya HTTP dan JSON. Web API memungkinkan aplikasi web untuk berkomunikasi satu sama lain melalui jaringan internet, sehingga memungkinkan interaksi antara aplikasi yang berjalan di berbagai perangkat dan *platform*. Web API biasanya digunakan untuk memungkinkan aplikasi web untuk terhubung ke layanan *cloud*, memanfaatkan data dari situs web lain, atau memungkinkan aplikasi *mobile* untuk terhubung ke aplikasi web.

2.12 PyQt

Qt merupakan *toolkit* yang digunakan dalam pembuatan aplikasi grafis lintas *platform* yang ditulis menggunakan bahasa pemrograman C++. *PyQt* memungkinkan untuk menggunakan *framework* GUI *Qt* dari *python*, sehingga dapat membuat aplikasi jauh lebih cepat tanpa mengurangi fitur *Qt* yang menggunakan bahasa C++.