PENGEMBANGAN SISTEM DETEKSI OBJEK PADA PRODUK RETAIL DENGAN ARSITEKTUR YOLOV4-*TINY*

Disusun Oleh:

N a m a : Raihan Digo Saputra

NIM : 19523235

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2023

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PENGEMBANGAN SISTEM DETEKSI OBJEK PADA PRODUK RETAIL DENGAN ARSITEKTUR YOLOV4-TINY

TUGAS AKHIR

Disusun Oleh:

N a m a : Raihan Digo Saputra

NIM : 19523235

Yogyakarta, 8 Juli 2023

Pembimbing,

(Dhomas Hatta Fudholi, S.T, M.Eng, Ph.D.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PENGEMBANGAN SISTEM DETEKSI OBJEK PADA PRODUK RETAIL DENGAN ARSITEKTUR YOLOV4-TINY

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 26 Juli 2023

Tim Penguji

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D.

Anggota 1

Rahadian Kurniawan, S.Kom., M.Kom.

Anggota 2

Zainudin Zukhri, S.T., MIT.

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dhomas Hatta Fudholi, S.T, M.Eng, Ph.D)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Raihan Digo Saputra

NIM : 19523235

Tugas akhir dengan judul:

PENGEMBANGAN SISTEM DETEKSI OBJEK PADA PRODUK RETAIL DENGAN ARSITEKTUR YOLOV4-*TINY*

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 7 Juli 2023

(Raihan Digo Saputra)

HALAMAN PERSEMBAHAN

Laporan tugas akhir ini dengan bangga peneliti persembahkan kepada keluarga besar peneliti terkhusus pakde, Almarhum bapak Eko Widodo, yang semasa hidupnya beliau selalu mendukung dan selalu memberikan masukkan kepada peneliti dalam pengerjaan laporan tugas akhir walaupun dalam keadaan sakitnya. Selain itu, peneliti juga persembahkan untuk diri peneliti sendiri yang telah berhasil menyusun laporan tugas akhir dengan berbagai keadaan peneliti.

HALAMAN MOTO

"Live as if you were to die tomorrow. Learn as if you were to live forever."

- Mahatma Gandhi

KATA PENGANTAR

Bismillahirrahmanirrahim.

Assalamu'alaikum Warahmatullahi Wabarakatuh

Segala puji serta syukur peneliti panjatkan kepada Allah SWT yang selalu memberikan rahmat serta hidayah kepada peneliti, sehingga dapat menyelesaikan laporan Tugas Akhir yang berjudul "Pengembangan Sistem Deteksi Objek Pada Produk Retail Dengan Arsitektur Yolov4-*tiny*". Tak lupa selawat beserta salam peneliti sampaikan kepada junjungan Nabi Muhammad SAW, sebagai suri tauladan bagi setiap umat manusia.

Laporan Tugas Akhir ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana (S1) Program Studi Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia. Peneliti memahami bahwa dalam penelitian laporan ini tidak terlepas dari bimbingan, doa, serta dukungan dari berbagai pihak. Pada kesempatan ini peneliti ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT yang selalu memberikan rezeki, dan rahmat-Nya sehingga peneliti dapat menyusun dan menyelesaikan laporan tugas akhir dengan baik.
2. Kedua orang tua bapak Zainal Abidin dan ibu Agustina Ribut Sudarmi, untuk segala doa dan dukungan kepada peneliti agar dapat menyelesaikan kuliah dengan baik.
3. Almarhum pakde Eko Widodo dan bude Dwi Hermawanti yang telah peneliti anggap sebagai orang tua kandung yang selalu memberikan dukungan dan doa kepada peneliti.
4. Keluarga yang memberikan dukungan, semangat, dan doa kepada peneliti.
5. Bapak Fathul Wahid, S.T., M.Sc., Ph.D., selaku Rektor Universitas Islam Indonesia.
6. Bapak Hari Purnomo, Prof., Dr., Ir., M.T., IPU, ASEAN.Eng., selaku Dekan Fakultas Teknologi Industri
7. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Ketua Jurusan Informatika Universitas Islam Indonesia.
8. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D., selaku Ketua Program Studi Informatika Program Sarjana Fakultas Teknologi Industri Universitas Islam Indonesia sekaligus dosen pembimbing tugas akhir yang telah membantu peneliti dalam memberikan arahan serta masukan dalam pengerjaan tugas akhir.
9. Bapak dan Ibu dosen Program Studi Informatika yang telah berjasa dalam memberikan ilmu yang bermanfaat kepada peneliti selama menuntut ilmu di Informatika UII.
10. Seluruh pihak yang tidak dapat peneliti sebutkan satu persatu, terima kasih atas semua dukungannya baik secara langsung ataupun tidak langsung.

Semoga seluruh ilmu, dukungan, bimbingan, dan doa, yang telah diberikan kepada peneliti mendapat balasan yang lebih baik dari Allah SWT. Peneliti menyadari bahwa laporan Tugas Akhir ini jauh dari kata sempurna dan memiliki kekurangan. Oleh karena itu, saran beserta kritik yang membangun sangat diharapkan. Peneliti berharap laporan Tugas Akhir ini dapat bermanfaat bagi semua pihak.

Yogyakarta, 07 Juli 2023

(Raihan Digo Saputra)

SARI

Supermarket sebagai toko retail yang menyediakan berbagai kebutuhan produk sehari-hari, dalam menjalankan proses bisnisnya, sering mengalami masalah kehabisan stok barang yang memberikan dampak kerugian paling banyak. Diperkirakan, kehabisan stok barang pada produk retail berkisar antara 7% hingga 10% di seluruh dunia, yang menyebabkan kerugian hingga miliaran tiap tahunnya. Masalah ini sebagian besar disebabkan oleh manajemen barang yang buruk. Adanya dukungan dari sistem deteksi objek diharapkan dapat meningkatkan kepedulian pihak manajemen terhadap pentingnya ketersediaan suatu barang sehingga proses jual beli pada toko retail menjadi lebih baik.

Berangkat dari permasalahan tersebut, pada penelitian ini peneliti menerapkan arsitektur model deteksi objek YOLO (*You Only Look Once*), yakni YOLOv4-*tiny* yang akan dijalankan pada perangkat *mobile.* YOLOv4-*tiny* digunakan karena memiliki berbagai kelebihan diantaranya terbukti memiliki kinerja yang baik, ukuran modelnya yang kecil, kompleksitas komputasi yang rendah, dan dapat memproses gambar dengan cepat. Selain itu, pada penelitian ini performa dari model ditentukan berdasarkan hasil nilai evaluasi *Mean Average Precision* (mAP). Adapun *dataset* yang digunakan pada penelitian ini sebagai studi kasus adalah produk susu bubuk sebanyak 4637 data gambar yang berisikan 106 kelas.

Selain itu, metodologi yang diterapkan dalam penelitian ini terdiri dari beberapa tahap meliputi analisis masalah, kalian literatur, *data collection, data preprocessing, data annotation, training model, evaluation,* dan *testing*. Proses *training* berhasil diselesaikan dalam waktu 111 jam, didapatkan hasil performa dari model yang dibangun cukup tinggi dengan nilai *mean average precision* sebesar 92,12%. Terlebih lagi, pengujian pada perangkat android mendapatkan waktu inferensi rata-rata berkisar antara 600 hingga 700ms.

Kata kunci: Kehabisan Stok Barang, Ketersediaan Barang Pada Rak, Deteksi Objek, YOLOv4-tiny, Produk Retail.

GLOSARIUM

Dataset	Kumpulan data yang digunakan pada penelitian.
Hyperparameter	Parameter yang disesuaikan untuk mengontrol proses pelatihan model pada pembelajaran mesin.
On-Shelf Availability	Ketersediaan suatu produk barang di rak toko.
Out-of-Stock	Kondisi suatu produk tidak tersedia atau habis persediaannya

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Supermarket yang dikenal dengan swalayan merupakan toko retail yang menawarkan kepada pelanggan berbagai macam produk kebutuhan (Puspa et al., 2020). Di Indonesia terdapat berbagai macam *supermarket* yang menguasai pasar penjualan produk retail diantaranya Super Indo, Hypermart, Carrefour, Lotte Mart, Giant, dan lain-lain (Angelia, 2022; Suryadarma et al., 2007). Saat menjalankan proses bisnis, *supermarket* ini tidak luput dari masalah-masalah yang menimbulkan kerugian. Masalah umum yang sering dijumpai pada *supermarket* diantaranya pencurian barang (Alfiah, 2021), barang di rak yang terjual habis (Hafiz Ar et al., 2021), dan barang yang sudah melewati batas waktu penggunaan atau basi (Disemandi & Ariesta Nadia, 2021). Dari masalah yang ada, kerugian paling banyak berasal dari produk yang mengalami OOS (*Out-of-Stock*) atau kehabisan barang (Campo et al., 2000).

OSA (*On-Shelf Availability*) merupakan suatu istilah untuk menggambarkan ketersediaan suatu produk barang di rak toko (Saragih et al., 2013). Selain itu, OSA menggambarkan tentang pelayanan dengan *Supply Chain* yang baik, yang memiliki perumpamaan jika suatu produk tidak tersedia maka produk tersebut tidak dijual (Wulandari et al., 2020). Adanya OSA (*On-Shelf Availability*) yang baik akan meningkatkan penjualan (Yilmazer & Birant, 2021).

OSA dapat menjadi OOS (*Out-of-Stock*) jika tidak diperkirakan dengan baik. Tingkat produk tidak tersedia atau OOS pada sektor grosir diperkirakan berkisar antara 7% hingga 10% di seluruh dunia dan sebanyak 70% hingga 75% disebabkan oleh manajemen barang yang buruk (Ranjan & Puri, 2012). Ketika hal tersebut terjadi, 31% pembeli akan membeli produk di toko lain, 26% membeli merek yang berbeda, 19% membeli produk yang sama dengan ukuran yang berbeda, 15% membeli produk pada lain waktu, dan 9% tidak jadi membeli (Corsten & Gruen, 2003; Mitchell, 2012). Hal tersebut tentu menjadi hal yang sangat penting untuk diatasi karena dapat mengurangi kerugian yang akan diterima.

Di Eropa tingkat produk tidak tersedia atau OOS tercatat sebanyak 8,3%, hal tersebut berdampak pada kerugian suatu industri hingga miliaran setiap tahunnya (Mitchell, 2012). Di Indonesia sendiri belum adanya data yang bersifat keseluruhan mengenai pentingnya ketersediaan barang. Akan tetapi, Berdasarkan penelitian yang telah dilakukan di Bantul,

Yogyakarta didapatkan data bahwa nilai koefisien regresi kelengkapan suatu produk sebesar 0.387 dengan nilai signifikansi sebesar 0,000 yang berarti 0,000 < 0,05. Oleh karena itu, dapat disimpulkan bahwa kelengkapan produk berpengaruh positif dan juga signifikan terhadap kepuasan konsumen (Anjarwan, 2018).

Sistem inventori memainkan peran sentral dalam mengatasi permasalahan kompleks yang dihadapi oleh toko-toko retail terkait ketersediaan produk. Pengelolaan stok produk yang efisien dan tepat waktu menjadi elemen krusial dalam meningkatkan kepuasan pelanggan dan meningkatkan efektivitas operasional toko Milella et al. (2020). Dalam banyak kasus, banyak toko retail masih menghadapi tantangan dalam mengelola inventori secara manual, yang seringkali menyebabkan ketidakakuratan (Muningsih & Kiswati, 2015) dan keterlambatan dalam pembaruan stok (Rumiarti et al., 2019).

Jika terjadi OOS (*Out-of-Stock*) pada produk retail akan menimbulkan reaksi negatif dari pelanggan sehingga menyebabkan kerugian (Ezhilkumar, 2020). Jika dilihat lebih dalam, bagi perusahaan retail, manufaktur, ataupun lainnya, persediaan merupakan faktor kunci yang sangat penting untuk entitas tersebut (Saresa et al., 2021). OOS dapat terjadi diantaranya disebabkan oleh manajemen *stock* yang tidak akurat (Muningsih & Kiswati, 2015), kesalahan pendataan (Hijriani et al., 2020), dan terlambat untuk melakukan *restock* produk (Rumiarti et al., 2019). Terlambat dalam melakukan *restock* merupakan hal yang umum terjadi, hal ini dikarenakan pada umumnya OSA diperiksa oleh karyawan secara manual di sebagian besar toko. Pendekatan ini dinilai tidak efektif dan berkelanjutan karena terus menerus yang membutuhkan usaha manusia (Yilmazer & Birant, 2021). Oleh karena itu, pengecekan ketersediaan produk di rak perlu dilakukan setiap saat agar tidak terlambat dalam melakukan *restock*.

Penerapan teknologi dalam industri retail saat ini telah begitu banyak digunakan terutama dalam hal mengantisipasi terjadinya OOS, teknologi-teknologi tersebut diantaranya: sistem informasi ketersediaan barang pada toko (Akbar & Juliastrioza, 2015)(Arandhea & Puspitasari, 2021; Limanto, 2018; Rahmasari, 2019; Swasono & Prastowo, 2021), penggunaan robot seluler yang mampu memetakan serta mendeteksi SOOS (*Shelf Out of Stock*) (Paolanti et al., 2019), dan *object detection-product recognition* (Ardiansyah et al., 2021; Higa & Iwamoto, 2019; Jha et al., 2022; Melek et al., 2019; Milella et al., 2020; Saqlain et al., 2022; Šećerović & Papić, 2018; Selvam et al., 2022; Sinha et al., 2022; Yilmazer & Birant, 2021). Salah satu metode yang dapat digunakan untuk menganalisis OSA pada produk retail adalah *object detection*. Adanya *object detection* dapat mengurangi beban kerja yang berhubungan dengan

manusia karena dapat dilakukan dengan otomatis secara *real-time* (Paolanti et al., 2019). OOS tidak selalu mengenai kehabisan barang, tapi dapat digunakan untuk mengisyaratkan manajemen barang yang kurang baik. Jika OOS pada produk retail terus dibiarkan terjadi akan berdampak pada loyalitas pelanggan turun (Yuswandi & Supriyanto, 2021) sehingga membuat penurunan pesanan atau penjualan (Son et al., 2019) yang mengakibatkan pendapatan tidak maksimal. Berdasarkan masalah tersebut, pengembangan sistem deteksi objek untuk menganalisis ketersediaan produk perlu dilakukan, karena selain dapat mengurangi beban kerja manusia (Paolanti et al., 2019), sistem ini juga dapat memaksimalkan keuntungan yang diperoleh dengan meminimalkan terjadinya OOS (Yilmazer & Birant, 2021).

Terdapat berbagai macam model algoritma yang dapat digunakan untuk melakukan *object detection,* salah satunya dengan YOLOv4-*tiny* (*You Only Look Once version 4*) yang diterapkan pada penelitian ini. YOLOv4-*tiny* memiliki berbagai kelebihan diantaranya memiliki kinerja yang baik dalam akurasi, ukuran modelnya yang kecil, kompleksitas komputasi yang rendah (Liu et al., 2022). Selain itu, terdapat penelitian yang dilakukan oleh Jannah & Sutanto (2022) yang menggunakan algoritma YOLOv4-*tiny* untuk melakukan deteksi rias adat nusantara. Hasil dari penelitian tersebut, algoritma YOLOv4-*tiny* terbukti mampu mendapatkan hasil rata-rata yang baik dengan akurasi 95,20% dan waktu deteksi selama 327ms. Algoritma ini melakukan *training* pada *dataset* gambar yang telah diberi label atau *class*. Selain itu, algoritma ini juga dapat berjalan di perangkat *mobile*. *Model mobile* dipilih karena memiliki berbagai keunggulan, diantaranya untuk menjaga privasi, tidak memerlukan koneksi internet karena tidak melakukan komunikasi dengan *server,* dan *response time* yang cepat karena langsung dijalankan pada perangkat (Dai et al., 2020).

Berdasarkan pemaparan mengenai data dan fakta tentang pentingnya menganalisis OSA pada produk retail, maka penelitian ini akan mengembangkan *object detection* dengan menerapkan arsitektur YOLOv4-*tiny* yang dapat berjalan secara *real-time.* Data produk retail yang digunakan pada penelitian ini merupakan produk susu bubuk sebanyak 4637 data gambar dan 106 *class data*. Adanya penelitian ini, diharapkan pihak manajemen toko retail dapat meningkatkan kesadaran akan pentingnya ketersediaan suatu produk di rak dalam upaya memaksimalkan keuntungan, sementara pihak produsen dapat memonitor ketersediaan produk-produk mereka di rak pada toko retail.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas, adanya kebutuhan untuk membangun sistem *object detection* guna menganalisis OSA (*On-Shelf Availability)* pada produk retail. Apakah *object detection* yang dikembangkan menggunakan model arsitektur YOLOv4-*tiny* dapat digunakan dengan baik untuk mendeteksi objek pada produk retail?

1.3 Batasan Masalah

Agar masalah yang diteliti tidak terlalu luas dan menyimpang maka dilakukan pembatasan masalah, batasan-batasan tersebut yaitu:

a. Produk retail yang menjadi objek penelitian ini dibatasi pada kategori susu bubuk.
b. Jumlah kelas yang akan digunakan dibatasi hingga 106 kelas.
c. Produk retail merupakan bentuk keluaran tahun 2022.
d. Performa dari model ditentukan berdasarkan hasil nilai evaluasi *Mean Average Precision* (*mAP*).

1.4 Tujuan Penelitian

Tujuan dari penelitian ini yaitu:

a. Mengembangkan sebuah sistem *object detection* dengan model arsitektur YOLOv4-*tiny* untuk mengatasi terjadinya OOS (*Out-of-Stock*) pada produk retail.
b. Melakukan pengujian sistem *object detection* yang dikembangkan untuk melihat performa yang dihasilkan.

1.5 Manfaat Penelitian

Manfaat yang didapatkan dengan adanya penelitian ini yaitu:

a. Mengurangi kemungkinan potensi kerugian akibat OOS (*Out-of-Stock*).
b. Mempermudah dalam melakukan pengecekan ketersediaan produk pada rak karena tidak perlu dilakukan pengecekan secara manual.
c. Meningkatkan kesadaran pihak manajemen toko retail akan pentingnya ketersediaan suatu produk di rak.
d. Mempermudah pihak produsen dalam memonitor ketersediaan produknya di rak.

1.6 Metodologi Penelitian

Metodologi yang diterapkan dalam penelitian ini, terdiri dari beberapa tahap yaitu:

a. Analisis Masalah, merupakan tahap pertama yang dilakukan pada penelitian ini. Tahap ini dilakukan untuk menganalisis masalah utama yang diangkat ke dalam penelitian.
b. Kajian Literatur, merupakan tahap mengkaji penelitian terdahulu, hal tersebut dilakukan untuk menghindari terjadinya plagiat serta menambah pemahaman mengenai tema yang diangkat. Tahap ini melibatkan membaca berbagai jurnal yang terkait dengan solusi yang ditawarkan.
c. *Data Collection,* pada tahap ini dilakukan proses pengumpulan data berupa gambar. Nantinya data gambar tersebut akan digunakan sebagai *dataset* dalam proses pengembangan sistem deteksi objek.
d. *Data Preprocessing* atau *Cleaning* yaitu, pengolahan yang dilakukan untuk memperkecil kemungkinan *noisy* pada data gambar. Hal ini dilakukan karena baik atau buruknya suatu *dataset* akan mempengaruhi kinerja dari model.
e. *Data Annotation* atau *labelling* merupakan proses pelabelan data gambar. Hal ini dilakukan untuk memberikan tanda terhadap objek yang ditelah ditentukan sebelum dilakukan pelatihan.
f. *Model Training,* merupakan tahap pelatihan model dengan *hyperparameter*-nya. Proses yang dilakukan pada tahap ini dimulai dari *split data* hingga menganalisis model-model yang telah didapatkan.
g. *Evaluation*, pada tahap ini akan ditentukan model mana yang akan digunakan dengan melihat performa model yang telah didapatkan.
h. *Testing*, pada tahap ini model yang telah dipilih akan digunakan pada perangkat *mobile* untuk melihat performa model secara keseluruhan.

1.7 Sistematika Penulisan

Sistematika penulisan dalam penyusunan laporan tugas akhir ini terdiri dari beberapa bab, yang mencakup gambaran dari keseluruhan masalah dan penyelesaiannya. Berikut sistematika penulisan yang terbagi dalam 5 bab :

BAB I PENDAHULUAN

Pada bab ini berisi pembahasan latar belakang masalah, rumusan masalah, Batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian dan sistematika penulisan.

BAB II LANDASAN TEORI

Pada bab ini akan dibahas mengenai teori dasar yang digunakan memiliki keterkaitan dengan sistem dalam penerapan deteksi objek untuk menganalisis ketersediaan produk retail serta tinjauan terhadap penelitian yang sudah pernah dilakukan, berhubungan dengan apa yang akan dirancang dan diimplementasikan.

BAB III METODOLOGI PENELITIAN

Bab ini berisikan tentang analisis masalah, kajian literatur, serta uraian mengenai pengembangan sistem yang meliputi *data collection*, *data cleaning*, *data annotating*, *model training*, *evaluation,* dan *Testing* untuk menganalisis OSA.

BAB IV HASIL DAN PEMBAHASAN

Bab ini akan dibahas tentang hasil dari implementasi sistem deteksi objek serta evaluasi pengujian sistem deteksi objek yang telah dilatih dengan model.

BAB V KESIMPULAN DAN SARAN

Bab ini merupakan bab terakhir yang berisikan kesimpulan dari semua proses yang telah dilakukan serta terdapat juga saran yang dapat dimanfaatkan untuk penelitian berikutnya.

BAB II
LANDASAN TEORI

Pada bab ini akan dibahas mengenai teori dasar yang digunakan serta memiliki keterkaitan dengan sistem dalam penerapan deteksi objek untuk menganalisis ketersediaan produk retail. Selain itu, terdapat pula tinjauan terhadap penelitian yang sudah pernah dilakukan, berhubungan dengan apa yang akan dirancang dan diimplementasikan.

2.1 Produk Retail

Produk retail merupakan suatu kata yang terdiri dari 2 suku kata "Produk" dan "Retail". Menurut KBBI (2016a) produk merupakan barang atau jasa yang dibuat dan ditambah guna atau nilainya dalam proses produksi dan menjadi hasil akhir dari proses produksi itu. Di sisi lain, kata "Retail" diambil dalam bahasa Prancis "Retailer" yang memiliki arti memecah, memotong sesuatu menjadi lebih kecil dari sebelumnya (Chaniago, 2021; Fauza, 2017). Menurut KBBI (2016b) retail adalah usaha bersama dalam bidang perniagaan dalam jumlah kecil kepada pengguna akhir atau eceran. Dari kedua pengertian di atas, dapat ditarik kesimpulan bahwa produk retail merupakan barang atau jasa hasil produksi dalam jumlah yang kecil atau eceran.

Produk retail dapat dengan mudah dijumpai pada *Supermarket* atau yang lebih dikenal dengan swalayan. *Supermarket* merupakan toko retail yang menawarkan pelanggan berbagai macam produk kebutuhan (Puspa et al., 2020). Di Indonesia sendiri telah ada berbagai macam *supermarket* yang menguasai pasar penjualan produk retail diantaranya Super Indo, Hypermart, Carrefour, Lotte Mart, Giant, dan lain-lain (Angelia, 2022; Suryadarma et al., 2007). Proses yang terjadi pada produk retail hingga ke tangan konsumen dapat dilihat melalui Gambar 2.1.

Gambar 2.1 Distribusi barang dari produsen ke konsumen

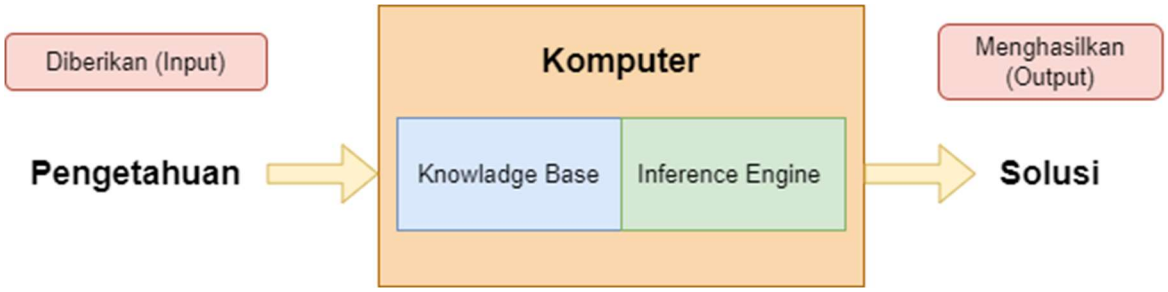
2.2 *On-Shelf Availability (OSA)*

On-Shelf Availability (OSA) atau dalam bahasa Indonesia dikenal dengan ketersediaan barang di rak merupakan suatu istilah untuk menggambarkan ketersediaan suatu produk barang di rak toko (Saragih et al., 2013). Ketersediaan produk merupakan suatu hal yang penting dalam menjalankan bisnis retail karena berpengaruh pada keputusan pembelian (Limanto et al., 2019).

Dalam serangkaian 29 studi di 20 negara yang melibatkan 71.000 konsumen, pada delapan kategori produk retail didapatkan hasil, jika terjadi kekosongan produk 31% pembeli akan membeli produk di toko lain, 26% membeli merek yang berbeda, 19% membeli produk yang sama dengan ukuran yang berbeda, 15% membeli produk pada lain waktu, dan 9% tidak jadi membeli (Corsten & Gruen, 2003; Mitchell, 2012). Dari data yang ada, sebanyak 40% pembeli tidak jadi membeli di toko tersebut, angka tersebut merupakan angka yang cukup tinggi untuk mempengaruhi penjualan dan profitabilitas. Ketersediaan produk di toko memiliki pengaruh positif terhadap keputusan pembelian. Hal ini berarti semakin stabil ketersediaan produk di toko maka akan semakin tinggi pula kesempatan pengambilan keputusan pembelian oleh konsumen (Saragih et al., 2013).

2.3 *Artificial Intelligence*

Artificial Intelligence atau yang lebih dikenal dengan kecerdasan buatan merupakan ilmu dan teknik untuk mengembangkan mesin cerdas, khususnya program komputer cerdas. Teknik ini menggunakan komputer untuk memahami kecerdasan manusia. Akan tetapi, AI tidak terbatas pada metode pengamatan biologis (Mccarthy, 2007). Pengembangan *artificial intelligence* terdapat dua bagian utama atau inti yaitu, *Knowledge Base* dan *Inference Engine*. *Knowledge base* atau basis pengetahuan terdiri dari berbagai fakta-fakta, pengetahuan, ataupun teori. Di sisi lain, *inference engine* merupakan kemampuan untuk menarik kesimpulan berdasarkan *knowledge base* yang ada (Amanda Putri, 2020). Proses ini digambarkan melalui Gambar 2.2 mengenai komponen dari *artificial intelligence*. Selain itu, *artificial intelligence* memiliki turunan yang ditunjukkan oleh Gambar 2.3.



Gambar 2.2 Komponen *artificial intelligence*

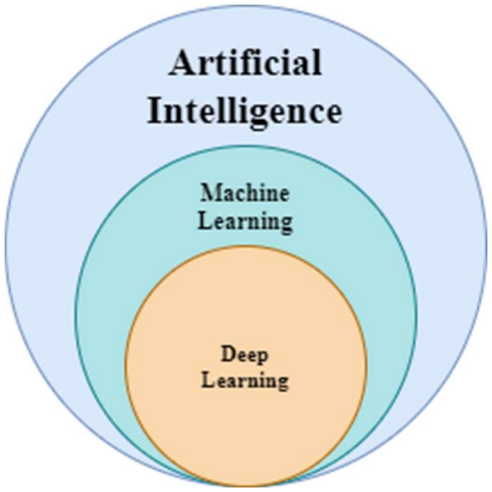
Sumber: Tjahyanti et al. (2022)

Tujuan dari dikembangkan *artificial intelligence* menurut Winston & Prendergast (1986) yang dikutip dari Gunarso (2013) antara lain untuk:

a. Tujuan ilmiah untuk memahami tentang kecerdasan.
b. Tujuan utama untuk membuat mesin jadi lebih pintar.
c. Tujuan *entrepreneurial* untuk membuat mesin menjadi lebih bermanfaat.

Penerapan *artificial intelligence* menurut Amrizal & Aini (2013) memiliki berbagai bidang terapan, yaitu:

a. Sistem pakar (*expert system*)
b. Pengenalan ucapan (*speech recognition*)
c. Jaringan saraf tiruan (*neural networks*)
d. Pengolahan bahasa alami (*natural language processing*)
e. Pengolahan citra (*image processing*)
f. mengakomodasi ketidakpastian (*Probabilistic reasoning*)
g. Game playing
h. Optimasi (*Evolutionary computing*)
i. *Robotics* dan *Sensory System*



Gambar 2.3 Hubungan antara *artificial intelligence, machine learning,* dan *deep learning*
Sumber: Jdid et al. (2021)

Berdasarkan Gambar 2.3 dapat diketahui bahwa *artificial intelligence* merupakan bidang yang lebih luas, mencangkup *machine learning* dan *deep learning.* Di sisi lain, konsep *deep learning* merupakan bagian dari *machine learning* yang lebih kompleks yang menggunakan jaringan syaraf tiruan untuk mempelajari suatu data.

2.4 *Machine Learning*

Machine learning merupakan turunan dari *Artificial Intelligence,* yang mana komputer dapat memiliki kemampuan untuk mempelajari secara "mandiri" tanpa harus diprogram ulang oleh manusia (Cholissodin et al., 2019). Alur proses *machine learning* dimulai dengan membangun suatu model berdasarkan *input* atau masukan sehingga dapat memberikan suatu prediksi atau pengambilan keputusan berdasarkan data, fakta, atau pengalaman yang ada. Terdapat empat kategori dalam pembelajaran *machine learning*, yaitu:

a. *Supervised Learning*

Supervised learning atau pembelajaran diawasi merupakan pembelajaran menggunakan data *input* yang telah diberi label/kelas yang menunjukkan kelompok dari data. Proses pembelajaran tersebut menghasilkan model dari data berlabel (Dewi, 2018).

b. *Unsupervised Learning*

Unsupervised learning atau pembelajaran yang tidak diawasi merupakan pembelajaran yang pada data belajarnya tidak diberi label sehingga perlu untuk mencari pola dari data

yang ada. Dari proses tersebut kemudian akan dilakukan pengelompokan berdasarkan karakteristik yang ditemui (Dewi, 2018).

c. *Semi-supervised Learning*

Semi-supervised learning menggunakan metode pembelajaran *unsupervised* untuk menemukan informasi tentang variabel *input*. Kemudian beralih ke *supervised* untuk memproses data dari kumpulan data yang tidak berlabel. Data diteruskan sebagai data pelatihan yang digunakan untuk membangun model prediksi baru dari masukan baru (Rahmatullah, 2022).

d. *Reinforcement Learning*

Reinforcement learning merupakan metode pembelajaran yang fase pembelajaranya dicampur dengan fase tes. Proses tersebut bertujuan memetakan situasi ke dalam tindakan untuk memperoleh penghargaan yang maksimal (Dewi, 2018).

2.5 *Deep Learning*

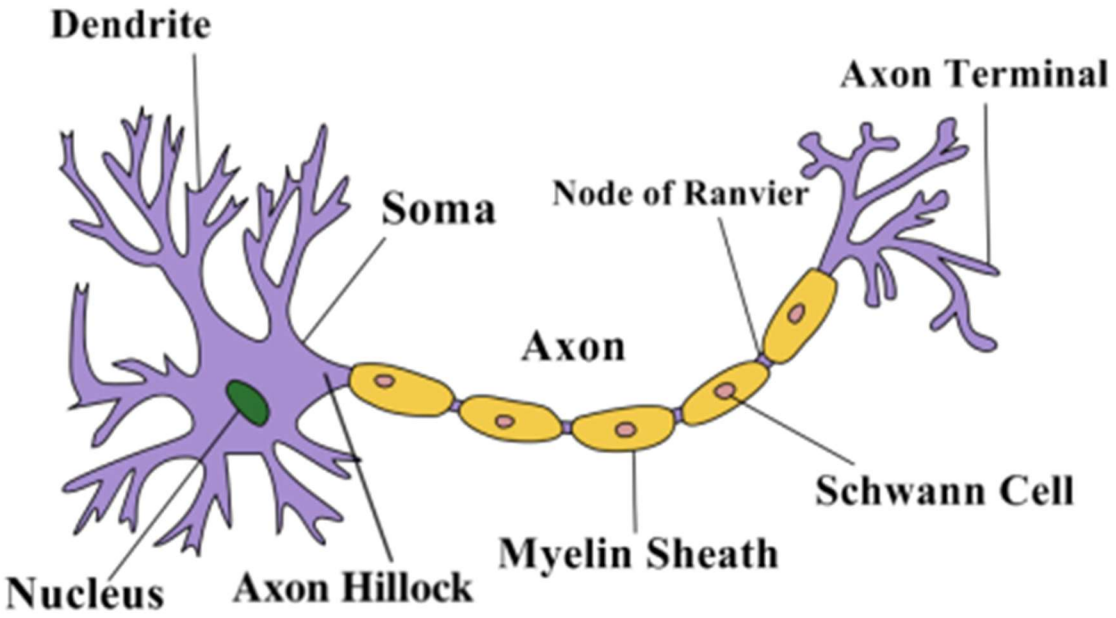
Deep Learning merupakan jenis *machine learning* yang mencapai fleksibilitas serta *power* yang sangat tinggi dengan mempelajari konsep hierarki (Goodfellow et al., 2016). Konsep *deep learning* membuat komputer mampu mempelajari konsep yang kompleks dari menggabungkan konsep yang sederhana. Penerapan *deep learning* pada komputer membuat komputer dapat mengklasifikasikan teks, suara, gambar, maupun video (Zulkiflie, 2021). Pengolahan citra akan sangat baik jika dilakukan pada *deep learning* (Marifatul Azizah et al., 2018). Pada beberapa kasus yang terjadi, penerapan *deep learning* akan mencapai akurasi yang tinggi bahkan melebihi manusia, hal tersebut dapat terjadi dikarenakan model dilatih dengan dipadukan arsitektur yang mengandung banyak lapisan *neural network* (Mubarok, 2019).

Pengklasifikasian data dalam *deep learning* menggunakan metode pendekatan, yaitu *Training* dan *Testing*. Pada *training* suatu model *deep learning* akan mempelajari fitur yang ada di dalam setiap data, hal tersebut dimaksudkan agar dapat membedakan suatu label dengan label lainnya. Namun, pada *testing* data yang diuji dapat dianalisis berdasarkan hasil data *training* sebelumnya (Marifatul Azizah et al., 2018).

2.6 *Artificial Neural Network*

Artificial neural network (ANN) atau jaringan syaraf tiruan merupakan suatu metode yang mencoba untuk menggabungkan komputer dengan kemampuan menyimpan dan memproses data dalam jumlah yang besar dengan otak manusia yang memiliki kemampuan

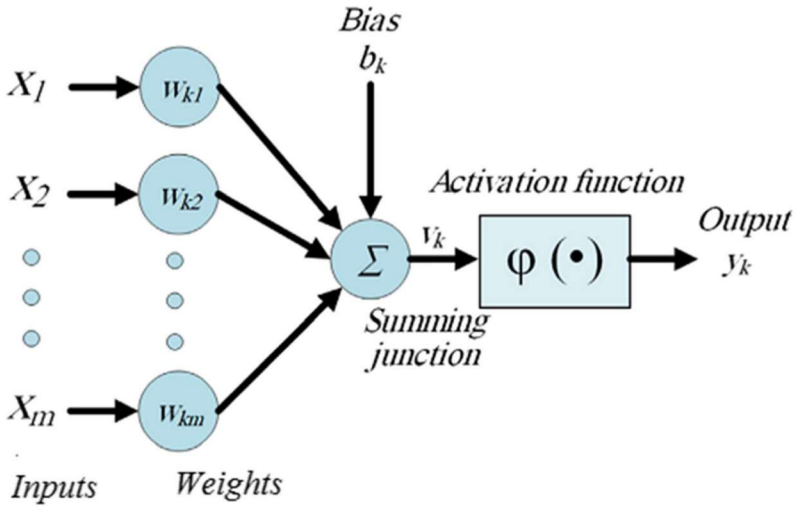
belajar (Mubarok, 2019). Hal tersebut diimplementasikan dengan cara meniru cara kerja sistem syaraf pada manusia yang diterapkan pada komputer, *node* yang dibangun terhubung satu sama lain, dengan *weight* atau bobot suatu *node* terikat melalui suatu *link* (Dewi, 2018). Di dalam otak manusia, terdapat ribuan hingga miliaran *neuron* yang saling terhubung, yang disebut *synapses*. Gambar 2.4 menunjukkan jaringan syaraf yang terdapat pada manusia.



Gambar 2.4 Neuron yang ada pada manusia

Sumber: Zhang (2019)

Sistem syaraf pada Gambar 2.4 bekerja dengan menerima sinyal dari *dendrit* kemudian diteruskan ke badan sel. Selanjutnya, sinyal diproses di badan sel dengan fungsi tertentu (*Summation* Proses). Jika sinyal tersebut melebihi ambang batas (*threshold*) maka sinyal akan membangkitkan *neuron* untuk melanjutkan sinyal. Sinyal akan dihilangkan jika sinyal tersebut berada di bawah ambang batas (*threshold*). Kemudian sinyal ditransmisikan menuju ke *axon* dan akhirnya ke *neuron* lain yang melewati *synapse* (Imantiyar, 2021). Gambar 2.5 merupakan gambar cara kerja jaringan syaraf tiruan.



Gambar 2.5 Sistem kerja jaringan syaraf tiruan

Sumber: Mohamed (2019)

2.7 *Image Processing*

Image processing atau dalam bahasa Indonesia dikenal dengan pengolahan citra merupakan proses mengubah suatu citra/gambar menjadi gambar lain dengan menggunakan teknik tertentu (Wells, 2014). Proses ini memiliki *input* dan *output* berupa citra/gambar. Pengolahan citra sering dijumpai pada bidang fotografi dalam pengubahan intensitas cahaya, perfilman dalam animasi, kedokteran dalam analisa medis (Dewi, 2018). Pengolahan ini memiliki tujuan untuk membuat kualitas citra/gambar menjadi lebih baik dengan memperbaiki kualitas citra manusia atau mesin lebih mudah dalam menginterpretasi (Nurkamid & Sutejo, 2010).

2.7.1 Prinsip Dasar Dalam Pengolahan Citra

Menurut Wells (2014) terdapat beberapa prinsip dasar dalam pengolahan citra, antara lain:

a. Peningkatan kecerahan dan kontras

Image enhancement merupakan suatu proses perbaikan citra dengan cara meningkatkan kualitas citra baik kontras maupun kecerahan (Nurkamid & Sutejo, 2010).

b. Penghilangan derau

Suatu citra ketika akan diproses seringkali mengandung terdistorsi atau mengandung *noise*, maka dari itu *noise* tersebut harus dihilangkan terlebih dahulu. Salah satu cara untuk mengatasi hal tersebut yaitu dengan filter *notch* (Wells, 2014).

c. Pencarian bentuk objek

Untuk mengidentifikasi objek dalam sebuah gambar, objek tersebut harus dipisahkan terlebih dahulu dari *background*-nya. Pendekatan umum yang digunakan untuk tujuan ini adalah deteksi batas objek yang mana batas objek merupakan tepi dari objek tersebut. Jika tepi objek diketahui, pencarian ciri objek dapat dilakukan (Wells, 2014).

2.7.2 Langkah Pengolahan Citra

Menurut Jalled & Voronkov (2016) pengolahan citra pada dasarnya mencakup tiga langkah berikut:

a. Mengimpor gambar dengan pemindai optik atau fotografi digital.
b. Menganalisis dan memanipulasi citra yang mencakup kompresi data dan peningkatan citra serta pola bercak yang tidak terlihat oleh mata manusia seperti foto satelit.
c. *Output* adalah tahap terakhir yang menghasilkan gambar atau laporan berdasarkan analisis gambar.

2.8 *Computer Vision*

Computer vision merupakan proses yang dihasilkan dari berbagai persepsi visual antara lain akuisisi citra, pemrosesan citra, pengenalan, serta pengambilan keputusan. Komputer dibuat meniru cara kerja indera penglihatan manusia yaitu mata (Wijaya & Prayudi, 2010). Untuk melakukan tugas tertentu pada visi komputer, suatu informasi dapat diambil dari berbagai format urutan video, pandangan beberapa kamera, bahkan data multidimensi dari pemindaian medis. *Computer vision* lebih cenderung untuk mempelajari bagaimana suatu komputer dapat mengenali suatu objek yang diamati (Fina et al., 2020). *Computer vision* secara umum merupakan kombinasi antara *image processing* atau pengolahan citra dengan *pattern recognition* atau pengenalan pola (Imantiyar, 2021). Menurut Dewi (2018) kemampuan visi komputer diantaranya terdiri dari:

a. *Object Detection*: proses mengenali suatu objek.
b. *Object Recognition*: proses melabeli suatu objek.
c. *Object Description*: proses memberikan properti kepada objek.
d. *3D Inference*: proses menafsirkan adegan 3D dari objek 2D yang dilihat.
e. *Interpreting Motion*: proses menafsirkan suatu gerakan.

2.9 *Object Detection*

Object detection atau yang lebih dikenal dengan deteksi objek dalam bahasa Indonesia merupakan suatu sistem yang berfungsi dalam mengenali sebuah objek dalam gambar ataupun video (Imantiyar, 2021). Secara teknis *object detection* merupakan suatu algoritma dalam *computer vision* yang menghasilkan kotak pembatas untuk kategori objek tertentu dan menetapkan skor untuk klasifikasinya (Bodla et al., 2017). Deteksi objek dibagi ke dalam dua bidang yaitu *soft detection*, yang hanya mendeteksi keberadaan objek, serta *hard detection* yang mendeteksi keberadaan dan lokasi dari suatu objek. Ditemukan fakta bahwa, dalam beberapa tahun terakhir data *training* dapat mempengaruhi kepekaan suatu model deteksi objek yang berbasis gambar (Jalled & Voronkov, 2016).

2.10 Tensorflow

Tensorflow merupakan suatu *machine learning system* yang beroperasi dalam skala yang besar dalam lingkungan yang heterogen yang bersifat *open source*, berdasarkan grafik komputasi (Abadi et al., 2016). Tim Google Brain dari organisasi penelitian Mesin Cerdas Google merupakan pengembang dari resmi dari Tensorflow. Tujuan umum dari sistem ini yaitu pembelajaran mesin dan penelitian jaringan syaraf dalam (Dewi, 2018). TensorFlow menggunakan grafik *dataflow* untuk merepresentasikan komputasi dalam algoritma dan status tempat algoritma beroperasi. Adanya Tensorflow, akan membantu pengembang dalam bereksperimen dengan pengoptimalan baru dan algoritma pelatihan. Hal tersebut dikarenakan Tensorflow memiliki keunggulan dapat mendukung berbagai aplikasi yang berfokus pada *training* dan inferensi pada *deep neural networks* (Abadi et al., 2016).

Menurut Dewi (2018) Tensorflow memiliki fitur utama yang meliputi:

a. Memiliki skalabilitas yang tinggi pada mesin dengan data yang besar.
b. Mendukung dalam teknik pembelajaran mesin dan juga *deep neural network*.
c. Melakukan definisi, perhitungan, serta pengoptimalan secara efisien melalui ekspresi matematis yang melibatkan *array* multidimensi (*tensor*).
d. Pemrosesan yang dilakukan GPU bersifat transparan dengan manajemen yang otomatis serta optimalisasi memori yang sama dengan data yang digunakan.

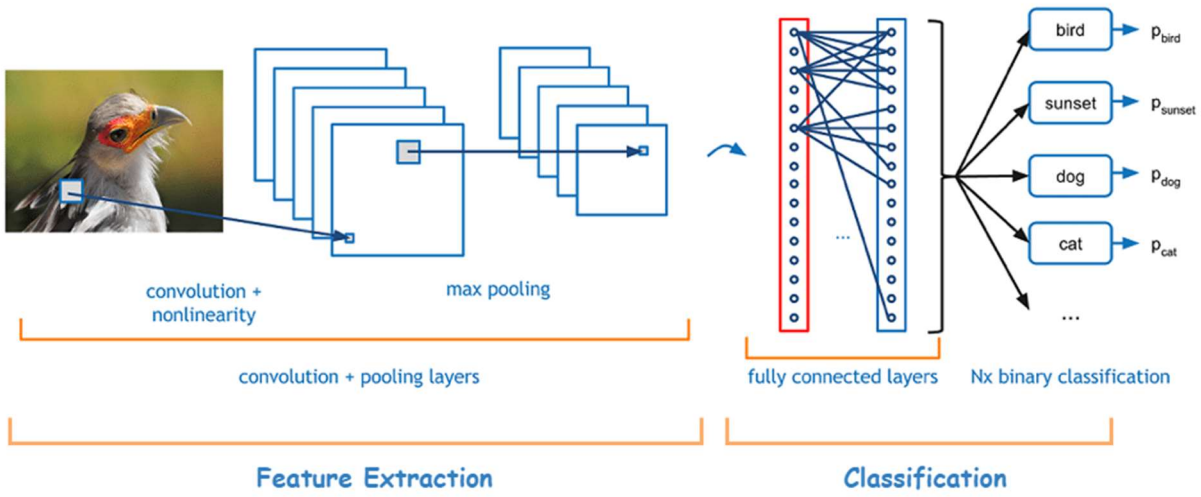
2.11 *Convolutional Neural Network (CNN)*

Convolutional Neural Network atau yang disingkat dengan CNN merupakan salah satu algoritma *deep learning* yang populer, algoritma ini dapat digunakan untuk melakukan

klasifikasi secara langsung pada bidang dua dimensi seperti teks, suara, gambar, ataupun video (Mubarok, 2019) tapi pada umumnya CNN digunakan untuk pemrosesan pada gambar (O'shea & Nash, 2015). Memiliki tingkat kedalaman jaringan yang tinggi membuat CNN termasuk ke dalam jenis *neural network* (Imantiyar, 2021). Pada Gambar 2.6 ditampilkan struktur dari *Convolutional Neural Network.*

Dikutip dari Zhiqiang & Jun, (2017):

> Pada umumnya CNN terdiri dari struktur lapisan fungsional yang berbeda, pada setiap tahapan berisikan *convolution layer, pooling layer,* dan *fully connected layer* yang berada dekat dengan *output*. *Training* pada CNN terutama menggunakan algoritma *forward propagation* dan *Back Propagation* untuk mempelajari *layer connection weights*, bias, dan parameter lainnya.



Gambar 2.6 Struktur dari *Convolutional Neural Network*

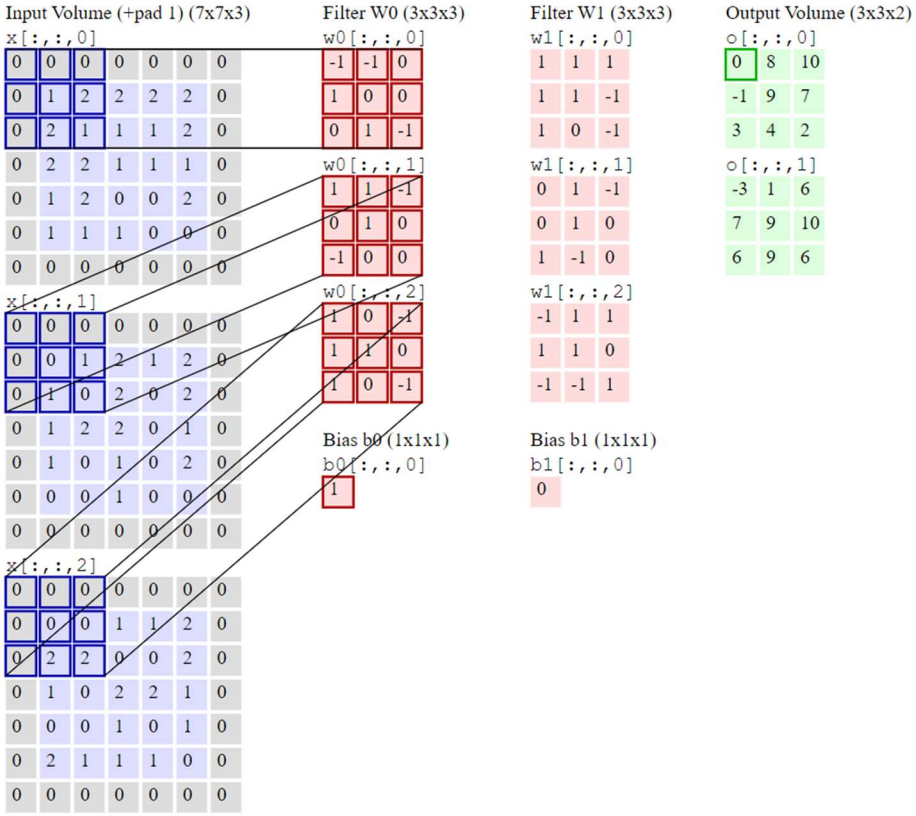
Sumber: Zhang (2019)

Gambar 2.6 menunjukkan bahwa arsitektur dari CNN dibagi ke dalam dua tahapan, yaitu *Feature extraction* dan *classification*. *Feature extraction* memiliki dua bagian utama, yaitu *convolutional layer* dan *pooling layer* yang banyaknya dapat disesuaikan dengan kebutuhan. Namun, pada *classification* hanya terdiri dari *fully connected layer.*

2.11.1 *Convolutional Layer*

Convolution layer atau dalam bahasa Indonesia disebut lapisan konvolusi merupakan lapisan yang akan mengolah *input* gambar/citra pertama kali. Selain mengolah citra untuk pertama kali, dalam kondisi tertentu *convolution layer* akan melakukan operasi konvolusi pada

output dari *layer* sebelumnya, hal ini disebabkan karena mengaplikasikan suatu fungsi pada output fungsi lain secara berulang (Zulkiflie, 2021). Lapisan ini dapat disebut sebagai lapisan inti, dikarenakan sebagian besar proses yang terjadi dilakukan pada lapisan ini (Firmansyah, 2021). *Convolution* dilakukan untuk mengekstraksi fitur dari citra/gambar *input* (Dewi, 2018). Proses ekstraksi fitur ditunjukkan pada Gambar 2.7.



Gambar 2.7 Proses yang terjadi pada *Convolutional Layer*

Sumber: Lina (2019)

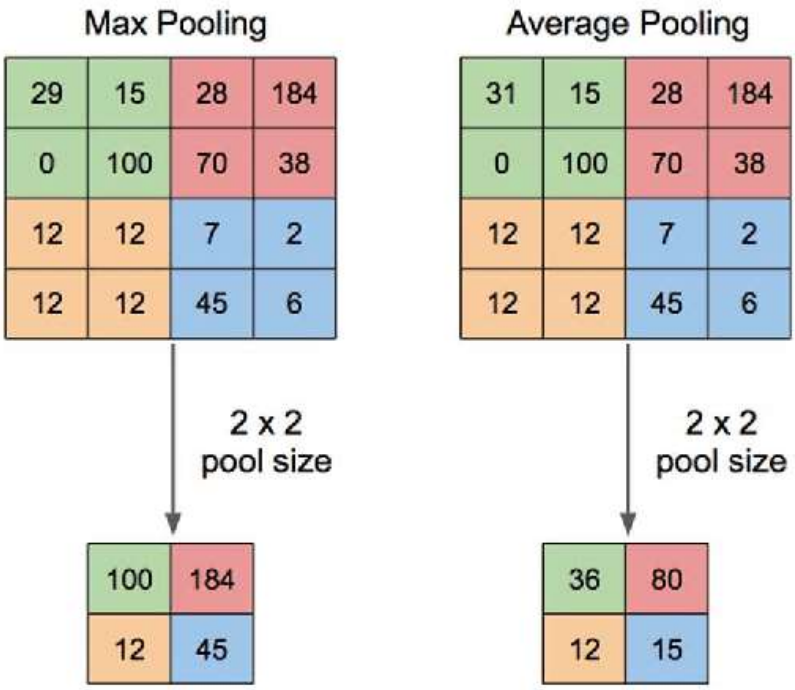
Gambar 2.7 merupakan gambaran dari proses yang terjadi pada *convolutional layer*. Filter akan melakukan pergeseran ke semua bagian gambar dari kiri atas hingga kanan bawah (Hijazi et al., 2015). Operasi "*dot*" dilakukan antara nilai *input* dengan nilai filter, ketika terjadi pergeseran filter. Hasil dari proses tersebut yaitu output berupa *feature maps* (Lina, 2019).

2.11.2 *Pooling Layer*

Pooling merupakan tahap untuk mengurangi data yang ada pada gambar dengan melakukan *down-sampling*. Dengan adanya tahap ini, data dapat direpresentasikan menjadi lebih kecil, mudah dikelola, dan mudah mengontrol *overfitting* tanpa menghilangkan data

pentingnya. Proses yang dikerjakan pada *pooling layer* ini yaitu *feature maps* (hasil dari *convolution layer*) setelah melewati *convolution function* dan *aktivasi* akan dilakukan *Pooling,* dengan cara menggeser bidang reseptif lokal pada gambar untuk menerapkan operasi yang dipilih (Firmansyah, 2021). Terdapat dua operasi yang dapat digunakan seperti pada Gambar 2.8 antara lain:

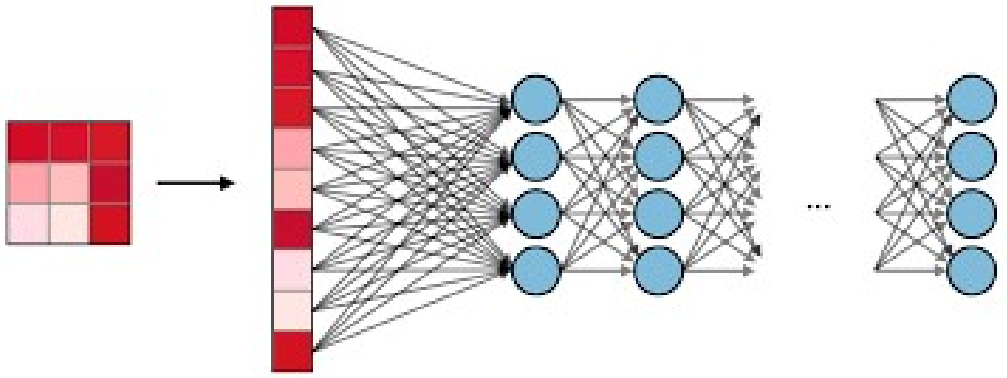
a. *Max Pooling*, merupakan cara memilih nilai yang tertinggi pada setiap bidang reseptif lokal.
b. *Average Pooling*, merupakan cara memilih nilai dengan menjumlahkan semua nilai pada bidang reseptif lokal kemudian dibagi banyaknya nilai pada bidang tersebut.



Gambar 2.8 Perbandingan operasi *Max Pooling* dan *Average Pooling*
Sumber: Yani et al. (2019)

2.11.3 *Fully Connected Layer*

Fully connected layer merupakan lapisan terakhir dari CNN. Setelah melewati *pooling* data akan digabung menjadi suatu vektor satu dimensi, hal tersebut dimaksudkan agar dapat diklasifikasikan dengan cara linear (Imantiyar, 2021). Vektor ini yang akan dijadikan *input* pada *fully connected layer*. *Output* dari *layer* ini didapatkan dengan melakukan penjumlahan *weight* atau bobot pada *input* dan respon dari *activation function* (Zhiqiang & Jun, 2017). Gambar 2.9 merupakan ilustrasi proses yang terjadi pada *fully connected layer*.

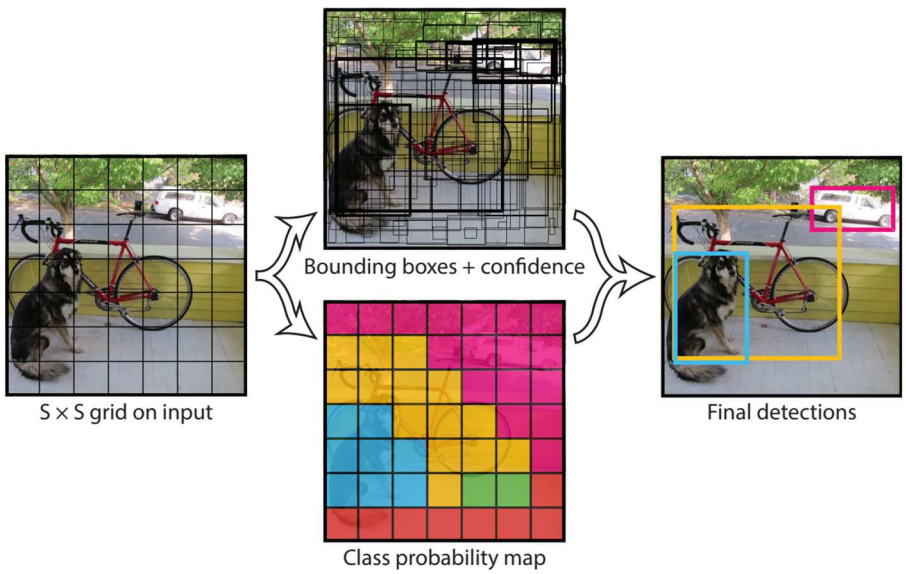


Gambar 2.9 Ilustrasi proses *Fully Connected Layer*

Sumber: Amidi & Amidi (2019)

2.12 *You Only Look Once (YOLO)*

YOLO atau *You Only Look Once* merupakan salah satu arsitektur dalam *object detection* yang cukup terkenal (P. Jiang et al., 2022). Arsitektur ini pertama kali diperkenalkan oleh Redmon et al. (2015). YOLO merupakan bagian dari *Convolutional Neural Network* (CNN) yang banyak diterapkan pada data gambar/citra (Khairunnas et al., 2021). YOLO juga merupakan "*One-stage Detector*" yang berbasis pada *deep learning* (Khatami, 2022). *One-stage Detector* memiliki karakteristik akurasi yang lebih rendah dengan kecepatan yang lebih cepat, maka dari itu layak untuk digunakan pada perangkat bergerak (He et al., 2019). Gambar 2.10 merupakan proses deteksi menggunakan arsitektur YOLO.



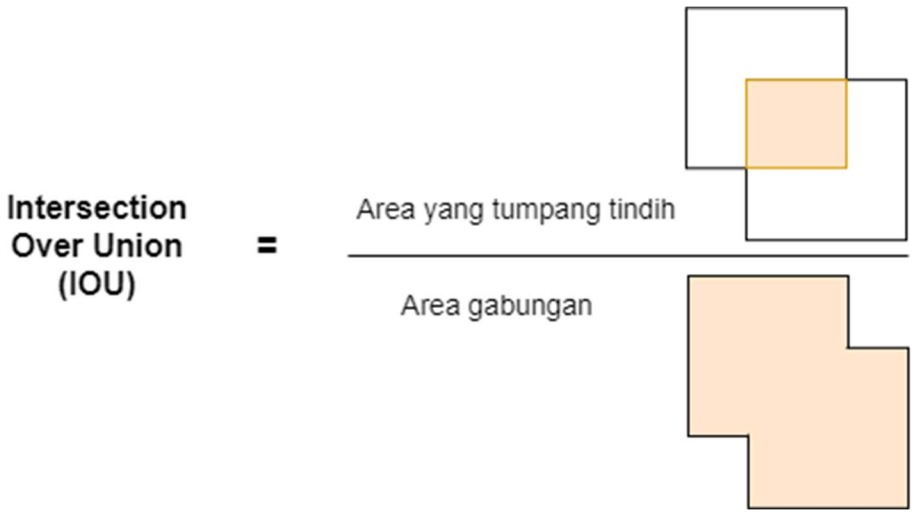
Gambar 2.10 Proses deteksi menggunakan arsitektur YOLO

Sumber: Redmon et al. (2015)

Berdasarkan Gambar 2.10 dan dikutip dari Redmon et al. (2015) cara kerja YOLO sebagai *object detector* dimulai ketika YOLO menerima suatu gambar sebagai *input,* kemudian gambar tersebut dibagi menjadi sel *grid* yang berukuran SxS. Jika titik *center* suatu objek berada di dalam sel *grid*, maka sel *grid* tersebut yang bertanggung jawab melakukan deteksi objek. Setiap sel grid memprediksi B *bounding boxes* dan *confidence score* dari tiap kotak. *Confidence score* menggambarkan seberapa yakin suatu model dengan keberadaan suatu objek di dalam kotak dan seberapa akurat juga prediksi tersebut. Akan tetapi, jika di dalam suatu kotak tersebut tidak terdapat objek maka nilai dari *confident score* menjadi nol. Hal tersebut didefinisikan melalui persamaan (2.1).

$$S_{conf} = Pr(Object) \, . \, IOU_{pred}^{truth} \tag{2.1}$$

Setiap *bounding box* terdiri dari lima nilai prediksi, yaitu x, y, w, h, dan *confidence score*. Pada nilai x dan y merupakan representasi dari titik pusat sel *grid*. Nilai w merupakan representasi dari lebar sel *grid* dan nilai h merupakan representasi dari tinggi sel *grid*. Nilai *confidence score* merepresentasikan *Intersection Over Union* (IOU) antara *predicted box* dengan *ground truth box*. Gambar 2.11 merupakan gambaran mengenai IOU.



Gambar 2.11 *Intersection Over Union*

Sumber: Padilla et al. (2020)

Pada Gambar 2.11 dapat dilihat bahwa nilai IOU (*Intersection Over Union*) sendiri didapatkan dari perbandingan area yang tumpang tindih antara *bounding box* dengan *ground*

box dan area gabungan antara *bounding box* dengan *ground box* (Rosebrock, 2016). Nilai ini dapat juga digunakan untuk menghitung *true positives* dan *false positives* pada serangkaian prediksi (Rezatofighi et al., 2019).

2.12.1 Keunggulan *YOLO*

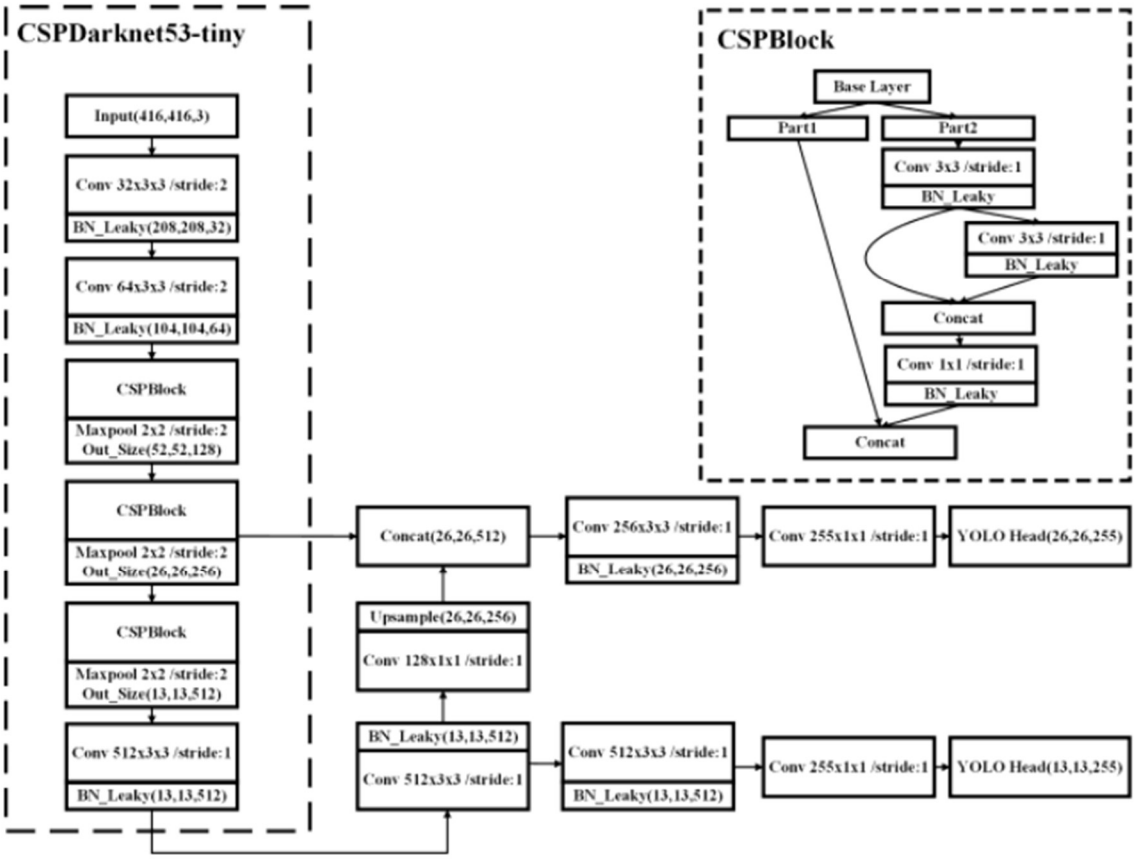
Berdasarkan penjabaran oleh Redmon et al. (2015) model arsitektur ini memiliki beberapa keunggulan dibandingkan dengan metode *object detection* tradisional, diantaranya:

a. Arsitektur YOLO dapat memproses gambar dengan sangat cepat, yaitu pada 45 hingga lebih dari 150 fps. Hal tersebut dapat membuat YOLO dapat berjalan secara *real-time* dengan latensi kurang dari 25 milidetik.
b. Arsitektur YOLO melihat seluruh gambar secara global pada saat membuat prediksi. Hal tersebut membuat YOLO menghasilkan kurang dari setengah jumlah kesalahan latar belakang dibandingkan dengan *Fast* R-CNN karena melihat konteks yang lebih besar.
c. Arsitektur YOLO mempelajari representasi objek yang dapat digeneralisasikan.

2.12.2 *YOLOv4-tiny*

YOLOv4-*tiny* merupakan arsitektur yang dirancang berdasarkan pada YOLOv4 dengan menyederhanakan struktur jaringan dan mengurangi parameternya untuk mendapatkan model arsitektur yang mampu mendeteksi objek dengan lebih cepat. Selain itu, Arsitektur ini merupakan salah satu model YOLO yang ringan. Arsitektur ini sangat dapat digunakan dalam *embedded system* ataupun perangkat bergerak (Z. Jiang et al., 2020).

Struktur arsitektur dari YOLOv4-*tiny* yaitu terdiri dari CSPDarknet53-*tiny* sebagai *backbone* jaringan, jaringan ini menggunakan CSPBlock untuk membagi *feature map* menjadi dua bagian dan menggabungkan keduanya dengan *cross stage residual edge*. Adanya modul CSPBlock terbukti dapat meningkatkan kemampuan belajar jaringan konvolusi (Z. Jiang et al., 2020). Pada bagian *neck* menggunakan struktur FPN yang dapat mengintegrasikan *feature* dari berbagai skala. Pada bagian *head*-nya arsitektur ini menghasilkan dua skala prediksi yaitu 26x26 dan 13x13. Hasil tersebut diperoleh dari *feature map* yang dihasilkan dari FPN. Proses prediksi yang dilakukan oleh arsitektur YOLOv4-*tiny* memiliki kesamaan dengan yang dimiliki YOLOv4 (Liu et al., 2022). Struktur dari arsitektur YOLOv4-tiny ditunjukkan pada Gambar 2.12.



Gambar 2.12 Struktur YOLOv4-*tiny*

Sumber: Z. Jiang et al. (2020)

Meskipun arsitektur ini merupakan turunan dari YOLOv4, YOLOv4-*tiny* memiliki berbagai kelebihan seperti yang dipaparkan oleh Liu et al. (2022), diantaranya:

a. Memiliki kinerja yang baik dalam akurasi.
b. Ukuran modelnya yang kecil.
c. Kompleksitas komputasi yang rendah.

2.13 *Model Mobile Machine Learning*

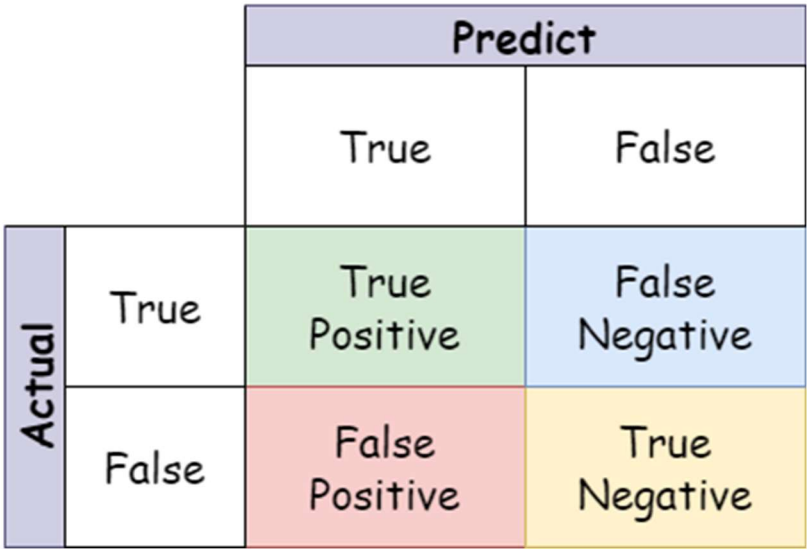
Model mobile Machine Learning merupakan suatu metode penerapan pembelajaran mesin ke dalam perangkat bergerak (Dai et al., 2020). Pada metode ini, model yang telah dilakukan pelatihan akan dimuat ke dalam perangkat *mobile* agar dapat digunakan. Dikutip dari Dai et al. (2020) *model mobile* memiliki berbagai keunggulan, diantaranya yaitu:

a. Privasi terjaga, hal tersebut dikarenakan proses yang dilakukan hanya terjadi di perangkat *mobile*.
b. Tidak memerlukan koneksi internet karena tidak melakukan komunikasi dengan *server.*

c. *Response time* yang cepat karena langsung dijalankan pada perangkat.

2.14 Pengukuran Performa

Perhitungan performa dari suatu arsitektur *machine learning* secara manual dapat dilakukan dengan bantuan *confusion matrix*. *Confusion matrix* merupakan suatu matriks yang mengklasifikasikan jumlah suatu data uji yang benar dan juga salah (Normawati & Prayogi, 2021). Pada Gambar 2.13 merupakan suatu tabel *confusion matrix* terdiri dari *True Positive (TP), False Negative (FN), False Positive (FP),* dan *True Negative (TN)* (Markoulidakis et al., 2021).



Gambar 2.13 *Confusion matrix untuk dua kelas data*

Sumber: Markoulidakis et al. (2021)

Keterangan:

a. *True Positive (TP)* = Banyaknya data kelas *true* diprediksi dengan benar sebagai kelas *true*.
b. *False Negative (FN)* = Banyaknya data kelas *true* diprediksi dengan salah sebagai kelas *false*.
c. *False Positive (FP)* = Banyaknya data kelas *false* diprediksi dengan salah sebagai kelas *true*.
d. *True Negative (TN)* = Banyaknya data kelas *false* diprediksi dengan benar sebagai kelas *false*.

Selain itu, dengan menggunakan *confusion matrix* dapat dihitung pula *precision, recall, f1-score,* dan juga *Average Precision* (Imantiyar, 2021; Naufal & Kusuma, 2021). Persamaan (2.2) hingga persamaan (2.4) dikutip dari penelitian Padilla, Netto, & Silva (2020) sedangkan persamaan (2.5) hingga persamaan (2.7) dikutip dari penelitian Harun et al. (2023).

Precision merupakan nilai kemampuan suatu model dalam melakukan identifikasi hanya pada objek yang relevan, rumus nilai *precision* ditunjukkan pada persamaan (2.2). *Recall* merupakan ukuran kemampuan dari suatu model untuk menemukan semua objek yang relevan, rumus nilai *precision* ditunjukkan pada persamaan (2.3). *F1-Score* dapat diartikan sebagai nilai gabungan antara nilai *precision* dengan *recall* (Tasnim et al., 2022), rumus nilai *F1-Score* ditunjukkan pada persamaan (2.4).

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.3}$$

$$F1Score = 2\,\frac{Precision.Recall}{Precision + Recall} \tag{2.4}$$

Setelah nilai *precision* dan *recall* telah didapatkan, berikutnya plot nilai *precision* terhadap *recall* untuk melihat pola yang berbentuk zig-zag. Pola zig-zag yang telah terbentuk akan dihaluskan, dimana nilai *precision* (P) pada setiap *recall* (r) diganti dengan nilai *precision* maksimum pada *recall* selanjutnya (r'). Perhitungan tersebut dapat dihitung dengan persamaan (2.5).

$$P_{inter}(r_{n+1}) = max\, P(r'); r' \geq r_{n+1} \tag{2.5}$$

Nilai AP (*Average Precision*) dihitung berdasarkan area yang berada tepat di bawah yang berbentuk kurva pada nilai *recall*, yaitu ketika nilai *precision* maksimum jatuh. Perhitungan nilai AP dapat dihitung dengan persamaan (2.6).

$$Average\ Precision = \Sigma(r_{n+1} - r_n)\, P_{inter}(r_{n+1}) \tag{2.6}$$

Mean Average Precision (mAP) merupakan hasil nilai rata-rata keakuratan dari suatu model dalam melakukan deteksi objek di semua *class*. Nilai N dalam persamaan (2.7) merujuk pada jumlah kelas yang digunakan pada model.

$$mAP = \sum_{i=1}^{N} \frac{AP_i}{N} x\ 100\% \tag{2.7}$$

2.15 *Studi Literatur*

Studi literatur merupakan upaya untuk memperoleh berbagai pengetahuan mengenai tema yang dibahas. Selain itu, Studi literatur juga dilakukan agar terhindar dari kegiatan plagiat. Hal yang dikaji berkaitan dengan masalah serta solusi yang dihadapi penelitian terdahulu, *dataset* yang digunakan pada saat melakukan penelitian, model arsitektur yang digunakan, serta performa dari arsitektur tersebut. Kajian literatur ini menggunakan 10 literatur yang berasal dari seluruh dunia, tidak hanya terbatas di Indonesia. Berikut beberapa ulasan literatur mengenai penerapan *object detection* pada produk retail.

Penelitian yang dilakukan oleh Saqlain et al. (2022), menyebutkan bahwa melakukan pemantauan pada rak secara terus menerus untuk manajemen ketersediaan barang merupakan faktor yang penting dalam meningkatkan penjualan serta kepuasan pelanggan. Dalam mengatasi masalah yang ada digunakan *deep learning* dan pendekatan *hybrid computer vision* yang disebut Hyb-SMPC. Hyb-SMPC terdiri dari dua modul, modul yang pertama bertujuan untuk mendeteksi produk ritel dengan menerapkan *one-stage deep learning*, yang membandingkan YOLOv4, YOLOv5, dan YOLO. Kemudian pada modul kedua dilakukan *planogram compliance*, dengan cara mengkonversi tata letak ke bentuk JSON lalu pencocokan dilakukan dengan gambar retail yang telah diproses. Penelitian ini menggunakan *dataset* yang telah disediakan mitra industri, *dataset* berisi produk dari berbagai kategori yang memiliki jumlah 30.000 gambar. *Dataset* diambil secara manual menggunakan kamera ponsel. Performa tertinggi dihasilkan oleh model YOLOv4 dengan *mAP* sebesar 0,96, *F1-score* 0,95, dan *recall* 0,898.

Selanjutnya, penelitian oleh Higa & Iwamoto (2019) dilakukan untuk memastikan ketersediaan rak yang tinggi. Hal ini merupakan upaya untuk mengurangi peluang penjualan yang hilang. Oleh karena itu, dilakukan pendeteksian terhadap produk yang kosong atau telah berpindah. Penelitian ini menggunakan *Convolutional Neural Network* (CNN) tanpa menggunakan *dataset* produk. Metode yang digunakan dengan cara mendeteksi wilayah dari

suatu produk yang berubah dengan menghilangkan *background,* diikuti dengan pemindahan objek yang bergerak. Hasil dari penelitian ini menunjukkan tingkat keberhasilan sebesar 89,6%.

Penelitian yang dilakukan Milella et al. (2020) menyebutkan bahwa pengelolaan ketersediaan dan inventaris yang efisien merupakan isu untuk mencapai kepuasan pelanggan dan mengurangi risiko kehilangan keuntungan. Solusi yang digunakan pada penelitian tersebut yaitu dengan menggunakan *depth sensor* yang memanfaatkan *modelling technique* dan rekonstruksi *3D point cloud*, serta pemasangan *occupancy grids* untuk memperkirakan ketersediaan produk. *Dataset* yang digunakan diperoleh dari titik penjualan eceran di Bari, Italia dengan pengambilan gambar dari atas rak produk. Arsitektur yang digunakan dalam penelitian ini yaitu *OSA estimation algorithm.* Hasil *maximum average error* yang didapatkan sebesar 5% yang diukur secara manual dan tingkat OSA lebih dari 0,98 di setiap *test case.*

Penelitian yang dilakukan Melek et al. (2019) menyebutkan bahwa dengan deteksi objek pada gambar di rak akan memecahkan berbagai permasalahan dalam penjualan retail. Solusi yang digunakan adalah dengan mengimplementasikan *deep learning*. *Dataset* yang digunakan pada penelitian ini yaitu *dataset* Coca-Cola dari ImageNet dan Grocery *dataset*. Dengan menerapkan arsitektur YOLOv2 sebagai modelnya, didapatkan performa terbaik pada iterasi ke-900 dengan *total loss* untuk *single class* sebesar 25,39% dan untuk *ten class* sebesar 29,84%.

Penelitian yang dilakukan Šećerović & Papić (2018) menyebutkan bahwa kehabisan stok merupakan masalah yang dihadapi semua toko retail. Untuk mengatasi masalah tersebut, Šećerović & Papić (2018) mengusulkan penggunaan kamera berukuran kecil yang dikombinasikan dengan *deep learning,* yaitu menerapkan arsitektur Faster R-CNN dan *Single Shot Multibox Detection* (SSD) dengan metode *transfer learning*. *Dataset* yang digunakan untuk merancang model tersebut adalah *custom dataset* yang terdiri dari 684 objek. Hasil performa yang didapatkan dari penelitian tersebut dibagi menjadi dua kategori, yaitu *object detection* dan *OOS detection*. Akurasi yang didapatkan pada kategori *object detection* dengan arsitektur *Faster* R-CNN sebesar 97,06% dan 75,42% pada arsitektur SSD. Di lain sisi, pada kategori *OOS detection* dengan menerapkan arsitektur *Faster* R-CNN diperoleh akurasi sebesar 55,55% dan 37,04% pada arsitektur SSD.

Penelitian yang dilakukan Ardiansyah et al. (2021) menyebutkan bahwa pemeriksaan barang secara manual merupakan aktivitas yang memerlukan biaya dan waktu yang besar. Solusi yang dikemukakan adalah dengan merubah pemeriksaan produk dari manual menjadi otomatis dan presisi. Penelitian ini menggunakan *dataset* yang diambil secara mandiri dengan

menggunakan kamera. Meskipun banyaknya *dataset* yang digunakan tidak dijelaskan, penelitian ini membagi gambar ke dalam tiga objek yang berbeda. Arsitektur yang diterapkan yaitu YOLOv3 dan berhasil mendapatkan performa sebesar lebih dari 90% untuk objek tunggal dan lebih dari 80% untuk objek jamak.

Penelitian yang dilakukan Yilmazer & Birant (2021) menyebutkan bahwa adanya ketersediaan barang yang tinggi merupakan faktor kunci untuk meningkatkan keuntungan. Solusi yang ditawarkan pada penelitian tersebut yaitu dengan menggunakan pendekatan *computer vision* untuk memantau ketersediaan produk di rak (OSA). Pada penelitian ini, *dataset* "WebMarket" digunakan sebanyak 3153 gambar yang diambil dengan menggunakan tiga kamera digital dengan jarak satu meter. Terdapat tiga *family* arsitektur yang diterapkan pada penelitian ini, yaitu YOLO, SOSA (*Semi-supervised learning On-shelf Availability*), dan RetinaNet. Pada *family* YOLO, hasil terbaik didapatkan pada YOLOv4 dengan mAP sebesar 0,91, *F1-score* sebesar 0,91, dan *Recall* 0,96. Pada *family* RetinaNet, hasil terbaik didapatkan dengan mAP sebesar 0,71, *F1-score* sebesar 0,74, dan *Recall* 0,82. Selain itu, penelitian ini juga menerapkan arsitektur SOSA dengan performa terbaik diperoleh mAP sebesar 0,89, *F1-score* sebesar 0,90, dan *Recall* sebesar 0,93.

Penelitian yang dilakukan Jha et al. (2022) menyebutkan bahwa pendekatan yang ada saat ini untuk mendeteksi kosongnya rak dan memastikan ketersediaan produk yang tinggi dinilai tidak efektif dan tidak layak karena manual, mahal, atau kurang akurat. Pada penelitian tersebut, solusi yang ditawarkan berupa pendekatan untuk merancang *pipeline machine learning* (ML) *end-to-end* untuk deteksi rak kosong secara *real-time*. *Dataset* yang digunakan pada penelitian tersebut merupakan *custom dataset* yang terdiri dari 1000 gambar produk di rak dari toko retail Target yang berlokasi di Amerika. Pada penelitian tersebut, arsitektur yang digunakan adalah beberapa rangkaian EfficientDet dan YOLOv5. Performa terbaik didapatkan pada model EfficientDet-D1 dengan *precision* sebesar 57,0%, *recall* sebesar 63,8%, *dan F1-score* sebesar 60,2% untuk *family model* EfficientDet dan pada model YOLOv5n6 didapatkan *precision* sebesar 76,3%, *recall* sebesar 63,8%, dan *F1-score* sebesar 71,3% untuk *family model* YOLOv5.

Penelitian yang dilakukan oleh Sinha et al. (2022) bertujuan untuk mengidentifikasi secara otomatis produk yang ditempatkan di rak toko retail. Solusi yang diberikan yaitu dengan menyempurnakan pendekatan yang sudah ada yakni *product recognition*. *Dataset* yang digunakan pada penelitian ini yaitu *dataset* Grozi-32k dan GP-180. Pada penelitian tersebut, arsitektur Faster R-CNN akan disempurnakan dengan mengembangkan pipa deteksi dan *two-*

step deep learning pipeline untuk mendeteksi lokasi produk serta mengidentifikasi produk yang ditempatkan di rak. Pada tahap pertama, Faster R-CNN-FPN akan digunakan dan pada tahap kedua, ResNet-18 akan digunakan. Hasil yang didapatkan dari penelitian tersebut yaitu model yang dilatih dengan *dataset* GP-180 menghasilkan mAP@0,5 sebesar 82,70 dan PR@0,5 sebesar 89,70, sedangkan untuk *dataset* Grozi didapatkan mAP sebesar 47,77% dan PR sebesar 55,11%.

Penelitian yang dilakukan oleh Selvam et al. (2022) menyebutkan bahwa mengaplikasikan deteksi *object* sudah banyak dilakukan, tapi hal tersebut dirasa masih kurang baik karena kemasan produk dapat berubah seiring waktu, dan produk yang berbeda terlihat sangat mirip. Solusi yang ditawarkan pada penelitian tersebut adalah dengan menerapkan dua metode, yaitu *product detection* dan *produk recognition,* yang terdiri dari *text detection* dan *text recognition*. *Dataset* yang digunakan pada penelitian ini cukup banyak karena mengembangkan arsitektur yang kompleks. *Dataset* tersebut terdiri dari dua jenis, yaitu *dataset text* dan *dataset* produk grosir. *Dataset* yang digunakan diantaranya GroZi-120, WebMarket, Grocery Products, Freiburg Groceries Dataset, ICDAR 2015, Total-Text, ICDAR2011, dan ICDAR2013. Arsitektur model yang dikembangkan pada penelitian ini yaitu pada *product detection*, digunakan YOLOv5 mencapai *precision* 89,4%, *recall* 88,2%, dan *f-measure* 86,26%. Di sisi lain, untuk *product recognition* menggunakan dua algoritma, yaitu TextSnake untuk *text detection* dan SCATTER untuk *text recognition,* dengan hasil performa yang didapatkan yaitu sebesar *precision* 92,1%, *recall* 86,8%, *dan f-measure* 83,31%.

Berdasarkan penelitian yang telah disebutkan sebelumnya, diketahui bahwa terdapat penelitian yang menerapkan arsitektur yang sama, yaitu penelitian dari Yilmazer & Birant (2021) dan Saqlain et al. (2022). Akan tetapi, kedua penelitian tersebut menggunakan arsitektur versi YOLOv4. Oleh karena itu, peneliti tertarik untuk mengembangkan sistem deteksi objek untuk menganalisa ketersediaan suatu barang menggunakan versi yang lebih terbaru dari arsitektur YOLOv4 yaitu YOLOv4-*tiny,* dengan menerapkan pada produk retail susu bubuk. Selain itu, belum ada penelitian ilmiah mengenai deteksi objek pada produk retail menggunakan arsitektur tersebut. Pada Tabel 2.1 merupakan perbandingan teknologi yang diterapkan pada penelitian terdahulu.

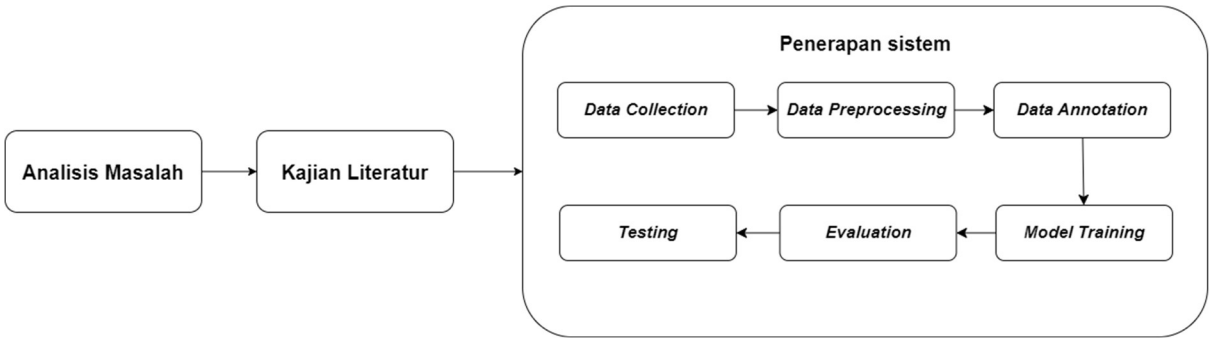
Tabel 2.1 Perbandingan Literatur

Penulis	Dataset	Model	Hasil
Saqlain et al. (2022)	*Dataset* disediakan mitra industri yang berisi produk dari berbagai kategori yang memiliki jumlah 30.000 gambar yang diambil secara manual melalui kamera ponsel	YOLOv4, YOLOv5, dan YOLOR	Performa tertinggi pada YOLOv4 dengan *mAP* sebesar 0,96, *F1-score* 0,95, dan *recall* 0,898.
Higa & Iwamoto (2019)	-	*Convolutional Neural Network* (CNN)	Tingkat keberhasilan sebesar 89,6%.
Milella et al. (2020)	*Dataset* yang diambil secara manual dititik penjualan eceran di Bari, Italia	*OSA estimation algorithm*	*Maximum average error* sebesar 5% diukur secara manual dan tingkat OSA lebih dari 0,98 di setiap *test case.*
Melek et al. (2019)	*dataset* Coca Cola dari ImageNet dan Grocery *dataset*	YOLOv2	Performa terbaik pada iterasi ke 900 dengan *total loss* untuk *single class* sebesar 25,39% dan untuk *ten class* sebesar 29,84%
Šećerović & Papić (2018)	*Dataset Common Objects in Context* (COCO)	Faster R-CNN dan *Single Shot Multibox Detection* (SSD)	Pada *object detection* dengan Faster R-CNN diperoleh 97,06% dan menggunakan SDD diperoleh 75,42%. pada *OOS detection* dengan Faster R-CNN diperoleh 55,55% dan menggunakan SSD 37,04%.
Ardiansyah et al. (2021)	*Dataset* yang diambil secara mandiri dengan menggunakan kamera, tapi banyaknya *dataset* yang digunakan tidak dijelaskan	YOLOv3	Performa sebesar lebih dari 90% untuk objek tunggal dan lebih dari 80% untuk objek jamak
Yilmazer & Birant (2021)	WebMarket *dataset*	YOLOv4 dan SOSA (*semi-supervised learning on-shelf availability*)	YOLOv4 memperoleh mAP 0,91, *F1-score* 0,91, dan *Recall* 0,96, sedangkan SOSA menghasilkan performa mAP 0,89, *F1-score* 0,90, dan *Recall* 0,93
Jha et al. (2022)	*Dataset* produk di rak dari toko retail Target yang berlokasi di Amerika	EfficientDet dan YOLOv5	YOLOv5n6 mencapai performa terbaik dengan *Validation* mAP 68,3, mAR 78,9, mAF 73,2 dan untuk *Test* memperoleh

Penulis	Dataset	Model	Hasil
			mAP 66,9, mAR 76,3, dan mAF 71,3
Sinha et al. (2022)	*Dataset* Grozi-32k dan GP-180	Pada tahap pertama digunakan Faster R-CNN-FPN dan pada tahap kedua digunakan ResNet-18	Hasil performa pada *dataset* GP-180 didapatkan mAP @ 0,5 sebesar 82,70 dan PR @ 0,5 sebesar 89,70, untuk *dataset* Grozi didapatkan mAP 47,77% dan PR 55,11%
Selvam et al. (2022)	GroZi-120, WebMarket, *Grocery Products*, ICDAR 2015, Total-*Text*, ICDAR2011, dan ICDAR2013	*Product detection* menggunakan YOLOv5, sedangkan *product recognition* menggunakan dua algoritma yaitu TextSnake untuk *text detection* dan SCATTER untuk *text recognition*.	*Product detection* mencapai *Precision 89,4%, Recall 88,2%,* dan *F-Measure 86,26%, s*edangkan *product recognition mencapai Precision 92,1%, Recall 86,8%, dan F-Measure 83,31%.*

BAB III
METODOLOGI PENELITIAN

Penelitian ini memiliki beberapa tahapan yang dilakukan oleh peneliti. Tahapan tersebut terdiri dari analisis masalah, studi literatur, *data collection, data preprocessing, data annotation, train model, evaluation,* dan *testing* seperti pada Gambar 3.1.



Gambar 3.1 Tahapan Penelitian

3.1 Analisis Masalah

Di Indonesia sudah banyak *supermarket* yang membuka cabang di setiap kota. *Supermarket* dimanfaatkan oleh masyarakat untuk mencari berbagai barang kebutuhan harian rumah tangga. Terdapat berbagai macam merek produk dan berat per sajian yang disediakan dalam *supermarket*. Hal tersebut tentu menjadikan *supermarket* sebagai tempat yang semakin penting bagi masyarakat (Iswandi & Ester, 2020). Proses pembelian suatu produk di *supermarket* secara umum dilakukan dengan pengambilan barang di rak secara mandiri lalu melakukan pembayaran di kasir telah disediakan. Ketika suatu produk diambil (dibeli) dari rak akan menyebabkan ketersediaan barang di rak menjadi berkurang. Jika tidak segera dilakukan pengisian ulang rak akan menjadi suatu masalah. Masalah tersebut menjadi masalah umum yang sering dijumpai pada *supermarket* yaitu barang yang ada pada rak habis (Hafiz Ar et al., 2021). Selain itu, masalah tersebut menjadi masalah yang menyebabkan kerugian paling banyak (Campo et al., 2000). Pada Gambar 3.2 merupakan keputusan yang terjadi jika suatu barang tidak tersedia.

Gambar 3.2 Data keputusan jika suatu barang tidak tersedia di toko

Gambar 3.2 menjelaskan respon yang diberikan pembeli ketika suatu barang tidak tersedia yaitu 31% membeli produk di toko lain, 26% membeli merek yang berbeda, 19% membeli produk yang sama dengan ukuran yang berbeda, 15% membeli produk pada lain waktu, dan 9% tidak jadi membeli (Corsten & Gruen, 2003; Mitchell, 2012). Selama ini proses pengecekan ketersediaan produk yang ada pada rak kebanyakan masih dilakukan secara manual. Hal tersebut tentu membutuhkan upaya manusia untuk memeriksa dan tidak efektif (Yilmazer & Birant, 2021).

Berakar dari masalah tersebut tema penelitian akan diangkat, yaitu mengenai penerapan sistem yang dapat mendeteksi objek produk retail untuk meningkatkan kesadaran pihak manajemen toko retail akan pentingnya ketersediaan suatu produk di rak. Dengan memanfaatkan *deep learning object detection* pemeriksaan barang dapat dilakukan secara *real-time* dan mengurangi beban kerja manusia.

3.2 Kajian Literatur

Pada tahap kajian literatur peneliti mengumpulkan berbagai penelitian terdahulu mengenai deteksi objek pada produk retail. Hal tersebut dilakukan untuk menghindari terjadinya plagiat. Literatur yang dikumpulkan setidaknya kurang dari 5 tahun agar pemahaman yang didapatkan merupakan pemahaman terbaru. Selain itu, literatur juga berasal berbagai negara untuk memperluas pemahaman peneliti mengenai teori yang berkaitan dengan tema penelitian. Literatur didapatkan dari berbagai situs penelitian diataranya Research Gate

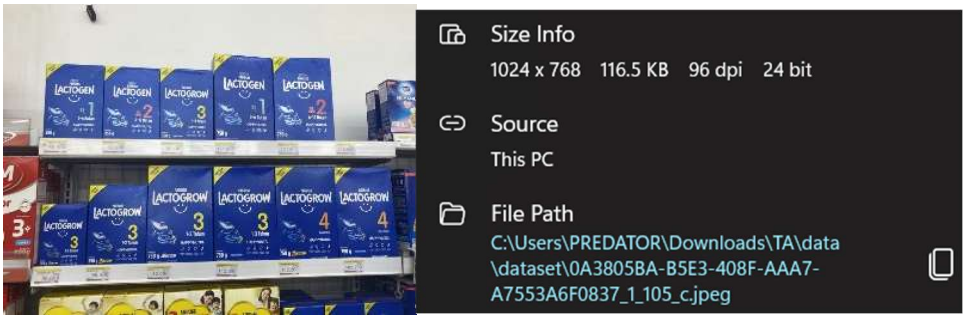
(www.researchgate.net), ScienceDirect (www.sciencedirect.com), Google Scholar (scholar.google.com), Academia Edu (www.academia.edu), dan IEEE Xplore (eeexplore.ieee.org)

3.3 *Data Collection*

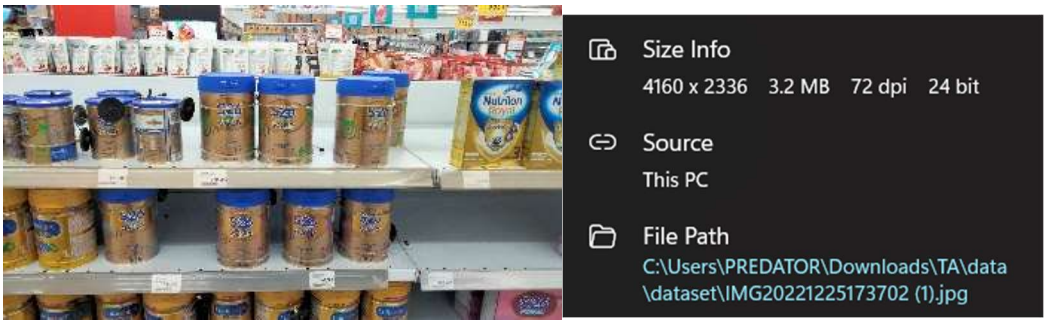
Pada tahap ini dilakukan proses pengumpulan *dataset* berupa data gambar. Data gambar yang berhasil dikumpulkan akan digunakan sebagai *dataset* dalam proses pengembangan sistem deteksi objek. Data gambar yang digunakan merupakan data produk retail susu bubuk baik dalam kemasan kaleng dan juga kemasan dus, data ini terdiri dari 4637 data gambar dengan 106 kelas produk. Produk susu dipilih karena memiliki keunikan tersendiri pada datanya. Keunikan ini terletak pada desain produk yang sama, tetapi memiliki berat yang berbeda. Pada Gambar 3.3 merupakan contoh data gambar yang akan digunakan untuk *object detection* pada penelitian ini.

Gambar 3.3 Data produk retail susu bubuk

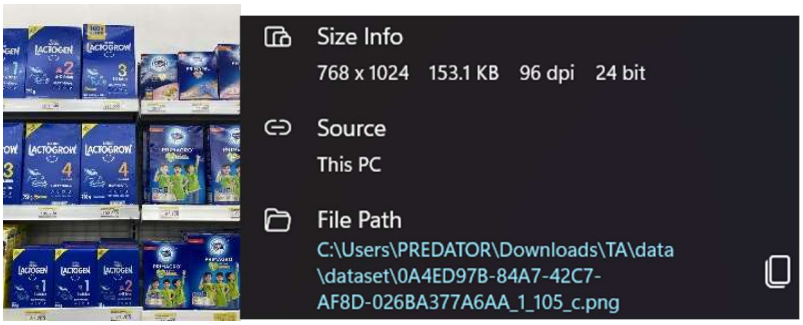
Berdasarkan Gambar 3.3 diketahui bahwa pada data gambar terdiri dari beberapa produk yang berbeda. Selain itu, Gambar 3.4 merupakan beragam resolusi yang ada pada *dataset* gambar tersebut, misalnya 1024x768 *pixels* ditunjukkan pada Gambar 3.4(a), 4160x2336 *pixels* ditunjukkan pada Gambar 3.4(b), dan 768x1024 *pixels* ditunjukkan pada Gambar 3.4(c). Selain itu, terdapat tiga format gambar yang ada pada *dataset,* yakni ".jpeg", ".jpg", dan ".png". Hal tersebut terjadi karena pengambilan gambar menggunakan beberapa perangkat. Luas pengambilan data gambar beragam mulai dari satu hingga lima rak seperti yang ditunjukkan pada Gambar 3.5.



(a) Gambar Beresolusi 1024x768 *pixels*



(b) Gambar Beresolusi 4160x2336 *pixels*



(c) Gambar Beresolusi 768x1024 *pixels*

Gambar 3.4 Ragam resolusi data gambar

(a) Luas satu rak

(b) Luas dua rak

(c) Luas tiga rak

(d) Luas empat rak

(e) Luas lima rak

Gambar 3.5 Luas pengambilan data gambar

Pada Gambar 3.5 dapat dilihat bahwa terdapat berbagai macam luas pengambilan data gambar mulai dari satu rak hingga lima rak, yang secara berurutan ditunjukkan pada Gambar 3.5(a) hingga Gambar 3.5(e). Selain itu, dapat diketahui pula bahwa semakin luas pengambilan

objek maka objek gambar akan semakin kecil. Jika suatu objek semakin kecil maka detail objek tersebut juga semakin tidak terlihat.

3.3.1 Populasi dan *Sample*

Populasi yang digunakan pada penelitian ini merupakan produk retail yang terdapat di toko, sedangkan *sample* ditentukan dengan menggunakan metode *non-probability convenience sampling* yang digunakan dalam penelitian ini yaitu 106 kelas produk susu bubuk. Metode *non-probability* digunakan karena ketidakpastian jumlah data populasinya, sedangkan *convenience sampling* digunakan karena pemilihan *sample* dapat disesuaikan dengan kebutuhan penelitian (Hasibuan, 2007). Total gambar produk retail susu bubuk yang digunakan 4637 gambar.

3.3.2 Jenis dan Sumber Data

Pada penelitian ini, peneliti menggunakan data sekunder, yang mana merupakan data yang sudah ada sebelumnya. Data ini bersumber dari data internal perusahaan yang bersifat *privacy* atau *confidential*. Data tersebut digunakan karena cocok dengan apa yang akan dilakukan pada penelitian ini yaitu meneliti tentang ketersediaan produk retail khususnya produk susu bubuk keluaran terbaru tahun 2022. Data produk susu diambil pada beberapa toko retail menggunakan perangkat *mobile*.

3.3.3 Kelas Data Gambar

Kelas produk yang digunakan pada penelitian ini berjumlah 106 kelas yang terdiri dari berbagai macam jenis produk susu bubuk. Kelas produk ini dapat dilihat pada Tabel 3.1.

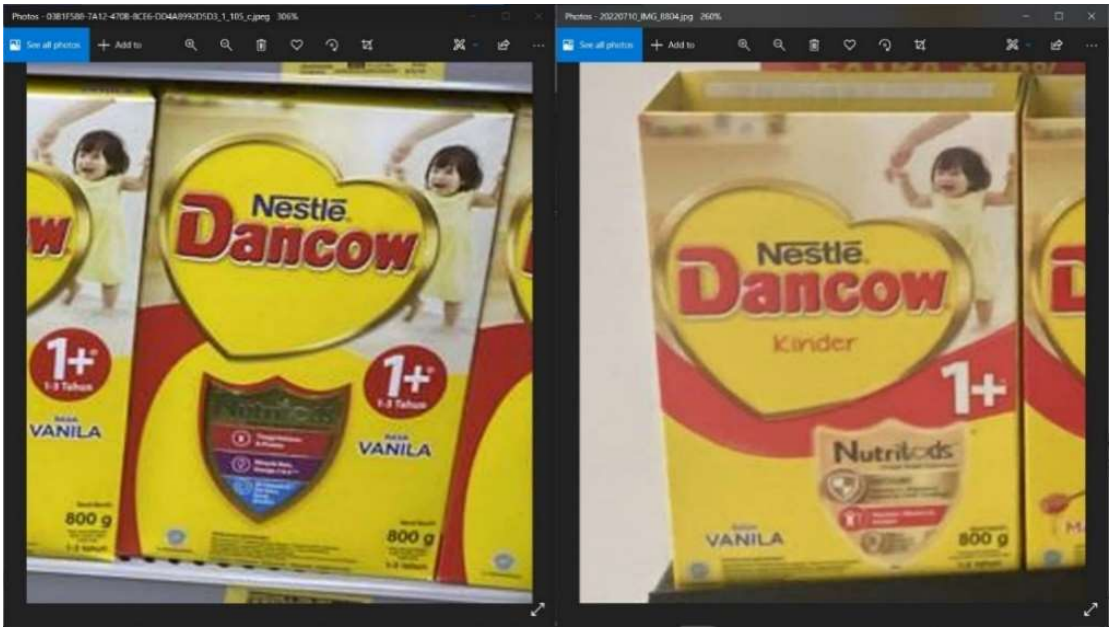
Tabel 3.1 Kelas produk susu

Kelas Produk Susu
BONEETO
DANCOW 1+ Cok Advn ExcNutr 12x800g
DANCOW 1+ Cok Advn ExcNutr 12x800g REDESIGN
DANCOW 1+ Cok Advn ExcNutr 24x400g
DANCOW 1+ Madu Advn ExcNutr 12x1000g
DANCOW 1+ Madu Advn ExcNutr 12x1000g REDESIGN
DANCOW 1+ Madu Advn ExcNutr 12x800g
DANCOW 1+ Madu Advn ExcNutr 12x800g REDESIGN
DANCOW 1+ Madu Advn ExcNutr 24x400g
DANCOW 1+ Madu Advn ExcNutr 24x400g REDESIGN
DANCOW 1+ Madu Advn ExcNutr 40x200g
DANCOW 1+ Madu Advn ExcNutr 40x200g REDESIGN
DANCOW 1+ Van Advn ExcNutr 12x1000g

Kelas Produk Susu
DANCOW 1+ Van Advn ExcNutr 12x1000g REDESIGN
DANCOW 1+ Van Advn ExcNutr 12x800g
DANCOW 1+ Van Advn ExcNutr 12x800g REDESIGN
DANCOW 1+ Van Advn ExcNutr 24x400g
DANCOW 1+ Van Advn ExcNutr 24x400g REDESIGN
DANCOW 1+ Van Advn ExcNutr 40x200g
DANCOW 1+ Van Advn ExcNutr 40x200g REDESIGN
DANCOW 3+ Cok Advn ExcNutr 12x800g
DANCOW 3+ Cok Advn ExcNutr 12x800g REDESIGN
DANCOW 3+ Cok Advn ExcNutr 24x400g
DANCOW 3+ Madu Advn ExcNutr 12x1000g
DANCOW 3+ Madu Advn ExcNutr 12x1000g REDESIGN
DANCOW 3+ Madu Advn ExcNutr 12x800g
DANCOW 3+ Madu Advn ExcNutr 12x800g REDESIGN
DANCOW 3+ Madu Advn ExcNutr 24x400g
DANCOW 3+ Madu Advn ExcNutr 24x400g REDESIGN
DANCOW 3+ Madu Advn ExcNutr 40x200g
DANCOW 3+ Madu Advn ExcNutr 40x200g REDESIGN
DANCOW 3+ Van Advn ExcNutr 12x1000g
DANCOW 3+ Van Advn ExcNutr 12x1000g REDESIGN
DANCOW 3+ Van Advn ExcNutr 12x800g
DANCOW 3+ Van Advn ExcNutr 12x800g REDESIGN
DANCOW 3+ Van Advn ExcNutr 24x400g
DANCOW 3+ Van Advn ExcNutr 24x400g REDESIGN
DANCOW 5+ Cok Advn ExcNutr 12x1000g
DANCOW 5+ Cok Advn ExcNutr 12x800g
DANCOW 5+ Cok Advn ExcNutr 12x800g REDESIGN
DANCOW 5+ Cok Advn ExcNutr 24x400g
DANCOW 5+ Madu Advn ExcNutr 12x1000g
DANCOW 5+ Madu Advn ExcNutr 12x1000g REDESIGN
DANCOW 5+ Madu Advn ExcNutr 12x800g
DANCOW 5+ Madu Advn ExcNutr 12x800g REDESIGN
DANCOW 5+ Madu Advn ExcNutr 24x400g
DANCOW 5+ Madu Advn ExcNutr 24x400g REDESIGN
DANCOW 5+ Van Advn ExcNutr 12x800g
DANCOW 5+ Van Advn ExcNutr 12x800g REDESIGN
DANCOW 5+ Van Advn ExcNutr 24x400g
DANCOW 5+ Van Advn ExcNutr 24x400g REDESIGN
ENFAGROW
FRISIAN FLAG
HILO
LACTOGEN 1 Happynutri 12x1kg
LACTOGEN 1 Happynutri 12x750g
LACTOGEN 1 Happynutri 24x350g
LACTOGEN 1 Happynutri 40x180g
LACTOGEN 2 Happynutri 12x1kg
LACTOGEN 2 Happynutri 12x750g
LACTOGEN 2 Happynutri 24x350g
LACTOGEN 2 Happynutri 40x180g
LACTOGROW 3 Happynutri 12x750g
LACTOGROW 3 Happynutri 24x350g

Kelas Produk Susu
LACTOGROW 3 Happynutri Honey 12x1kg
LACTOGROW 3 Happynutri Honey 12x750g
LACTOGROW 3 Happynutri Honey 24x350g
LACTOGROW 3 Happynutri Honey 40x180g
LACTOGROW 3 Happynutri Van 12x1kg
LACTOGROW 3 Happynutri Van 12x1kg REDESIGN
LACTOGROW 3 Happynutri Van 12x750g
LACTOGROW 3 Happynutri Van 24x350g
LACTOGROW 4 Happynutri Honey12x1kg
LACTOGROW 4 Happynutri Honey12x750g
LACTOGROW 4 Happynutri Van 12x1kg
LACTOGROW 4 Happynutri Van 12x1kg REDESIGN
LACTOGROW 4 Happynutri Van 12x750g
MORINAGA
NESTLE BATITA 1+ Madu+Iron 12x900g
NESTLE BATITA 1+ Madu+Iron 24x400g
NESTLE BATITA 1+ Madu+Iron 40x150g
NESTLE BATITA 1+ Vanilla+Iron 24x400g
NESTLE BATITA 1+Vanilla+Iron 12x900g
NESTLE DATITA 3+ Madu+Iron 12x900g
NESTLE DATITA 3+ Madu+Iron 24x400g
NESTLE DATITA 3+ Madu+Iron 40x150g
NESTLE DATITA 3+ Vanilla+Iron 24x400g
NESTLE DATITA 3+Vanilla+Iron 12x900g
NESTLE DATITA 5+ Madu+Iron 12x900g
NUTRICIA
PROVITAL
S-26 GOLD PROMIL 1 Can Top 6x900g
S-26 GOLD PROMIL 2 Can Top 6x900g
S-26 PROCAL GOLD Can 6x1,6kg
S-26 PROCAL GOLD Can Top 6x900g
S-26 PROCAL Pouch 12x400g
S-26 PROCAL Pouch 12x700g
S-26 PROCAL Pouch 6,2x700g
S-26 PROMIL 1 Pouch 12x400g
S-26 PROMIL 2 Pouch 12x400g
S-26 PROMISE GOLD Can 6x1,6kg
S-26 PROMISE GOLD Can Top 6x900g
S-26 PROMISE Pouch 12x400g
S-26 PROMISE Pouch 12x700g
SGM
VIDORAN

Dari beberapa kelas tersebut terdapat kelas yang memiliki kesamaan antara yang satu dengan yang lainnya. Hal tersebut disebabkan karena desain dari susu bubuk yang berubah contohnya seperti pada Gambar 3.6. Gambar sebelah kiri memiliki kelas “DANCOW 1+ Van Advn ExcNutr 12x800g”, sedangkan gambar sebelah kanan memiliki kelas “DANCOW 1+ Van Advn ExcNutr 12x800g REDESIGN”.



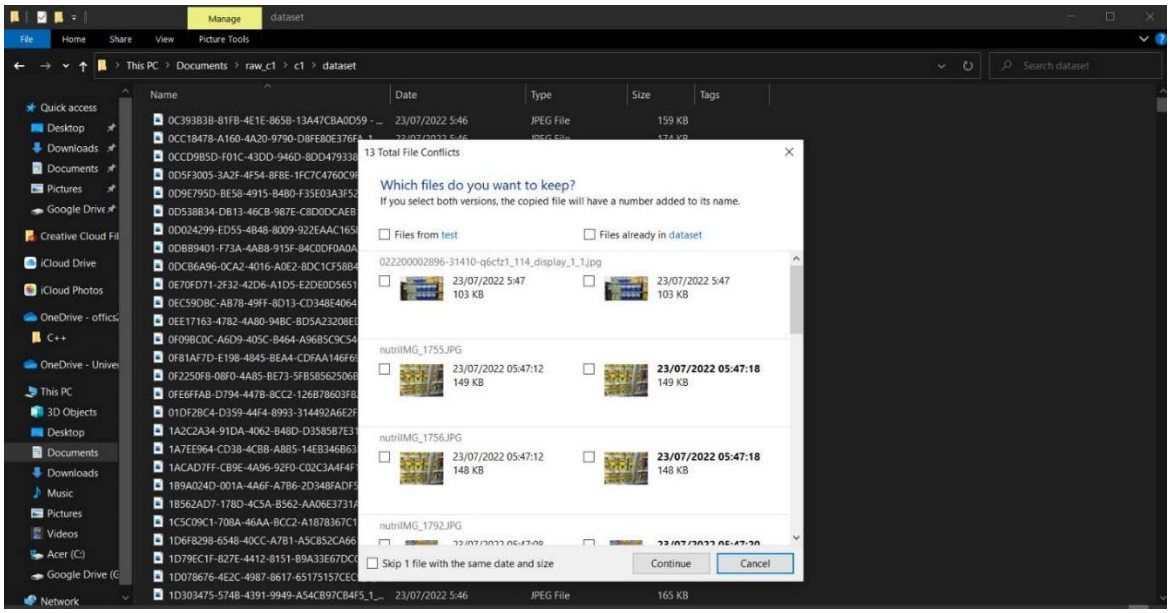
Gambar 3.6 Perbandingan antara dua kelas yang hampir mirip

3.4 *Data Preprocessing*

Pada tahap ini peneliti menggunakan data gambar yang telah dikumpulkan, lalu dilakukan pengolahan untuk memperkecil kemungkinan *noisy* pada data gambar. Memperkecil *noisy* dapat dilakukan dengan cara membuang data gambar yang terjadi duplikat dan melakukan *data preprocessing*.

3.4.1 Menghilangkan Gambar Duplikat

Data gambar yang memiliki kesamaan akan dilakukan penghapusan. Hal ini bertujuan untuk menghindari terjadinya bias dalam *dataset* serta gambar duplikat akan mempengaruhi kemampuan suatu model dalam melakukan generalisasi gambar baru. Proses ini dilakukan secara manual pada perangkat penelitian dengan mencari nama *file* yang sama. Proses ini ditunjukkan pada Gambar 3.7.



Gambar 3.7 Proses penghapusan data duplikat

Dari Gambar 3.7 diketahui terdapat 13 gambar yang memiliki nama *file* sama. Setelah dilakukan penghapusan data gambar, data akhir yang akan digunakan sebanyak 4637 gambar produk retail susu.

3.4.2 *Preprocessing Data*

Pada tahap ini data yang telah dibersihkan dari duplikat akan dilakukan *preprocessing* berupa penyesuaian resolusi, *sharpening*, dan penyelarasan format gambar. Penyesuaian resolusi dilakukan untuk menyamakan resolusi dari semua data gambar, yang semula terdapat berbagai macam resolusi akan dilakukan pengubahan menjadi 416x416 *pixels*. Pengubahan tersebut dilakukan karena model akan dilatih menggunakan jaringan 416x416. Selain itu, pengubahan resolusi juga dapat mengurangi *size* pada *dataset,* besar atau kecilnya ukuran ini berpengaruh pada waktu yang dibutuhkan untuk *train model*.

Pada tahap *sharpening* akan dilakukan untuk memperjelas atau mempertajam objek yang terdapat pada gambar. Proses penajaman gambar menggunakan bantuan dari *library* PIL (*Python Imaging Library*). *Library* ini merupakan *library* yang dapat digunakan untuk memanipulasi gambar (Gujar et al., 2016). Pada YOLOv4-*tiny* format gambar yang digunakan untuk melakukan pelatihan jaringan, yaitu berformat ".jpg" maka semua gambar perlu diubah ke dalam format ".jpg". Pada Gambar 3.8 merupakan kode program yang digunakan untuk melakukan *preprocessing* gambar.

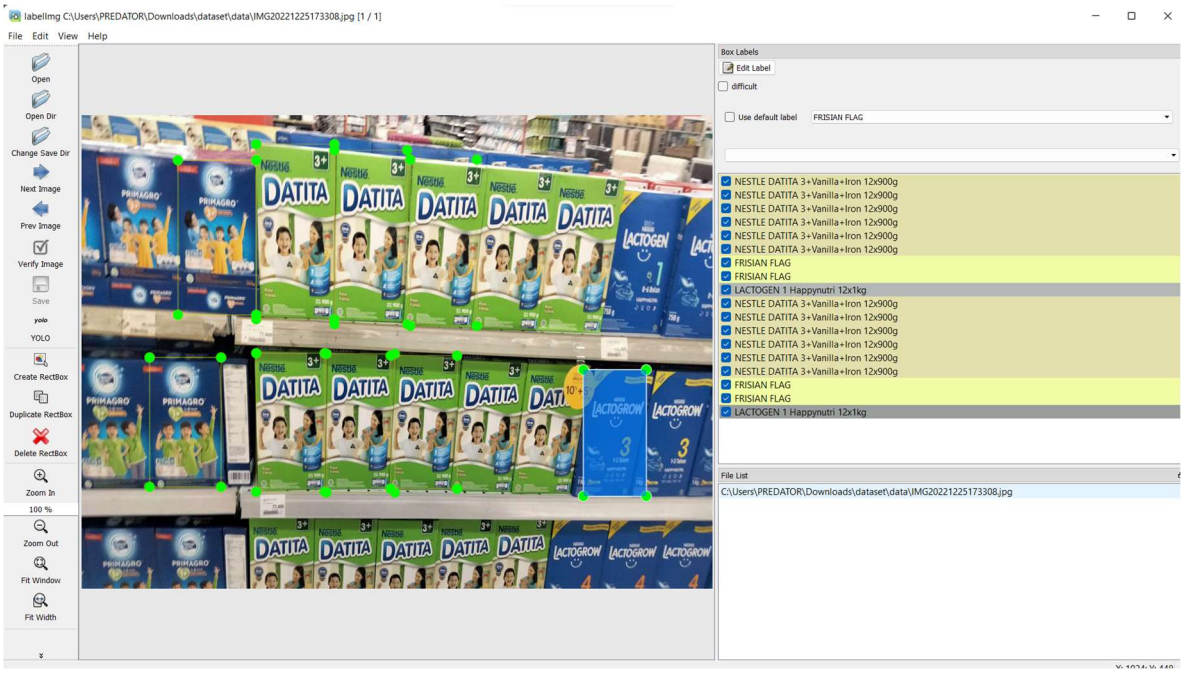
```
import os
import os.path
from PIL import Image
from PIL import ImageFilter

f = r'C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet\data\dataset'
for file in os.listdir(f):
    if (file.endswith(".jpeg")):
        f_img = f+"/"+file
        file_name = os.path.splitext(file)[0]
        img = Image.open(f_img)
        img = img.filter(ImageFilter.SHARPEN)
        img = img.resize((416,416))
        img.save("C:/Users/PREDATOR/Documents/Tugas Akhir/Program/Project TA 2/darknet/data/dataset/" + file_name +
".jpg")

for file in os.listdir(f):
    if (file.endswith(".jpeg")):
        f_img = f+"/"+file
        os.remove(f_img)
```

Gambar 3.8 Kode program proses *preprocessing*

Gambar 3.8 merupakan kode program yang digunakan untuk menjalankan proses *preprocessing*. Empat baris pertama pada program merupakan proses *import library* yang dibutuhkan untuk proses *preprocessing*. Pada baris ke-12 merupakan fungsi untuk menerapkan *sharpening* pada gambar. Baris ke-13 merupakan fungsi untuk melakukan pengubahan resolusi pada gambar. Pada baris berikutnya yaitu baris ke-14, data yang telah diubah akan disimpan ke dalam format ".jpg". Pada fungsi perulangan yang terakhir merupakan fungsi untuk menghapus data gambar yang sudah tidak digunakan lagi.

3.5 *Data Annotation*

Pada tahap ini data gambar yang telah dilakukan *preprocessing* akan dilakukan pelabelan, hal tersebut bertujuan untuk memberikan keterangan kategori atau *class object* sebelum dilakukan *model training*. Proses pelabelan dilakukan dengan bantuan *tools* "LabelImg" pada satu per satu data gambar, proses ini yang dapat dilihat pada Gambar 3.9.



Gambar 3.9 Proses *data annotation* menggunakan *tools* LabelImg

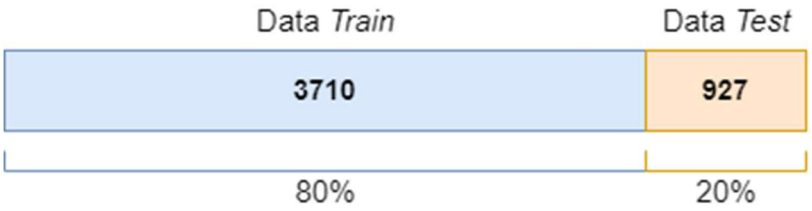
Gambar 3.9 merupakan proses pelabelan pada gambar dengan *tools* LabelImg. Proses pelabelan dilakukan dengan cara memberikan "RectBox" pada setiap sisi produk yang ingin diberikan label. "RectBox" ini berbentuk persegi yang akan menjadi pembatas atau penanda pada setiap produk, kemudian objek yang telah diberi penanda akan diberikan *class* dari objek tersebut yang berupa label berbagai macam produk susu bubuk. Pemberian label harus dilakukan dengan baik karena akan berpengaruh pada akurasi dari model deteksi objek. Selanjutnya, informasi dari data gambar yang telah diberi label akan disimpan dalam *file* ".txt" dengan format data YOLO.

3.6 *Model Training*

Pada tahapan ini data gambar yang telah dilakukan pelabelan atau *annotation* akan dilakukan pelatihan dengan model arsitektur YOLOv4-*tiny* serta konfigurasi *hyperparameter*-nya. Hal tersebut dimaksudkan agar model dapat mempelajari suatu pola atau ciri dari data gambar yang telah ada. Dari proses ini akan menghasilkan model yang dapat melakukan deteksi pada suatu objek khususnya dalam hal ini produk susu bubuk dengan tingkat akurasi yang tinggi serta dapat diterapkan pada perangkat bergerak.

3.6.1 *Split Dataset*

Sebelum data gambar dilakukan *train model*, data gambar terlebih dahulu harus dilakukan pemisahan atau *split* menjadi dua bagian yaitu *data train* dan *data test*. *Data train* digunakan untuk melatih suatu model, sedangkan *data test* digunakan untuk menguji performa dari suatu model. Pemisahan ini dilakukan dengan rasio perbandingan populasi 80% untuk *data train* dan 20% untuk *data test*. Sebanyak 3710 gambar akan digunakan sebagai *data train* dan sebanyak 927 gambar sebagai *data test*-nya. Pada Gambar 3.10 merupakan perbandingan antara *data train* dengan *data test.*



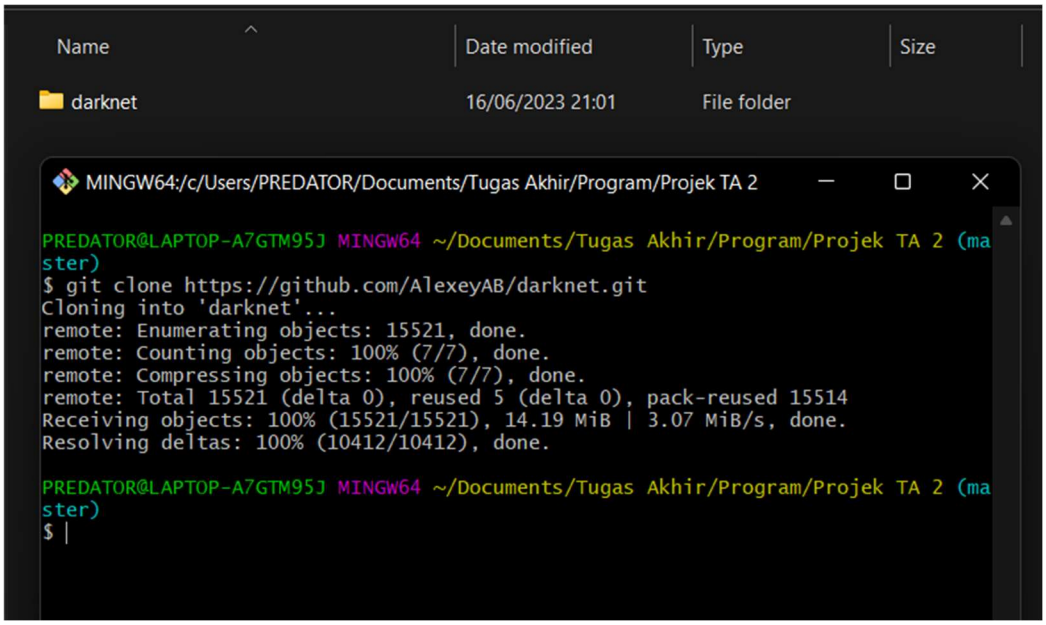
Gambar 3.10 *Split dataset*

Pada proses *split dataset* akan menghasilkan dua *file* yaitu "*train*.txt" dan "*test*.txt". *File* "*train*.txt" berisikan *path* semua data *train*, sedangkan "*test*.txt" yang berisikan *path* semua data *test.*

3.6.2 *Train Model*

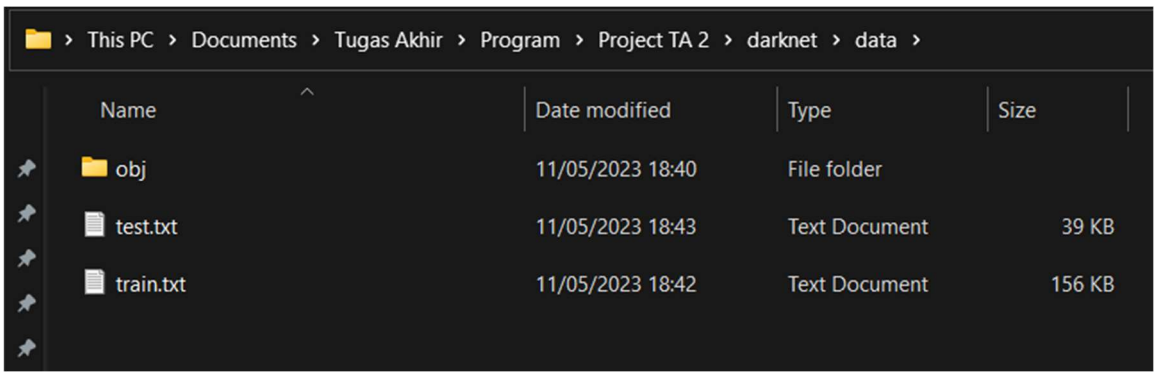
Arsitektur yang diterapkan pada penelitian ini yaitu YOLOv4-*tiny*. Model ini dipilih karena memiliki berbagai kelebihan, seperti yang dijabarkan oleh Liu et al. (2022), yakni diantaranya ukuran model yang kecil, tetapi memiliki kinerja yang baik. Hal tersebut merupakan faktor kunci dalam pengembangan *object detection* pada perangkat *mobile*. Proses pelatihan arsitektur YOLOv4-*tiny* akan dilakukan dengan menggunakan *framework* Darknet yang dikembangkan oleh Bochkovskiy (2021).

Hal pertama yang perlu untuk dilakukan yaitu melakukan proses *clone* terhadap *framework* Darknet. Proses *clone* ditunjukkan pada Gambar 3.11.



Gambar 3.11 *Clone repository* Darknet

Setelah melakukan *clone* seperti pada Gambar 3.11, hal yang berikutnya dilakukan yaitu menghapus semua *file* yang ada di *folder* data pada *directory* "../darknet/data/", hal ini bertujuan untuk mempermudah dalam melakukan konfigurasi pada model. Berikutnya pada *directory* ini juga akan dibuat *folder* baru yang bernama "obj". Tujuan dari dibuatnya *folder* ini, yaitu sebagai tempat penyimpanan *dataset* hasil dari proses *data annotation*. Selanjutnya, semua *dataset* tersebut dipindahkan ke *folder* tersebut. Pada Gambar 3.12 merupakan hasil pemindahan *dataset*.



Gambar 3.12 Pemindahan *dataset*

Gambar 3.12 menunjukkan hasil pemindahan *dataset*. Selain itu, dapat dilihat pula *file* hasil *split dataset* yaitu "*train*.txt" dan "*test*.txt" juga ikut dipindahkan. Hal tersebut dilakukan karena proses pelatihan akan dilakukan di dalam *folder* Darknet. Setelah *dataset* berhasil

dipindahkan, berikutnya dilakukan konfigurasi *hyperparameter* arsitektur YOLOv4-*tiny*. Konfigurasi *hyperparameter* disesuaikan dengan jumlah *class* yang digunakan, pada penelitian ini menggunakan 106 *class*. *File* yang dikonfigurasi dapat dijumpai pada *directory* "../darknet/cfg/yolov4-tiny-custom.cfg". Pada *file* tersebut terdapat banyak *hyperparameter* yang digunakan pada arsitektur YOLOv4-*tiny,* tetapi hanya delapan yang dilakukan konfigurasi. Hal tersebut mengacu pada penelitian yang dilakukan oleh Badharudheen (2021) terbukti mampu menghasilkan akurasi yang cukup baik. Konfigurasi dapat dilihat pada Tabel 3.2.

Tabel 3.2 Konfigurasi model YOLOv4-*tiny*

Parameter	Nilai
Batch	64
Subdivisions	32
Max_batches	212000
Steps	169600, 190800
[convolutional - pertama*]*	
filters	333
[yolo - pertama*]*	
classes	106
[convolutional - kedua*]*	
filters	333
[yolo - kedua*]*	
classes	106

Keterangan dikutip dari penelitian Badharudheen (2021):

a. *Batch* yang digunakan sebesar 64 dengan *subdivisions* sebanyak 32, hal tersebut dapat diartikan satu iterasi terdiri dari 64 gambar dengan 2 gambar per *mini-batch* yang akan diproses.

b. *Max_batches* diganti mengikuti persamaan (3.1).

$$Max\ batches = \ class.2000 \tag{3.1}$$

c. *Nilai steps* diganti mengikuti persamaan (3.2) dan (3.3).

$$Steps_1 = Max\ batches\ .\frac{80}{100} \tag{3.2}$$

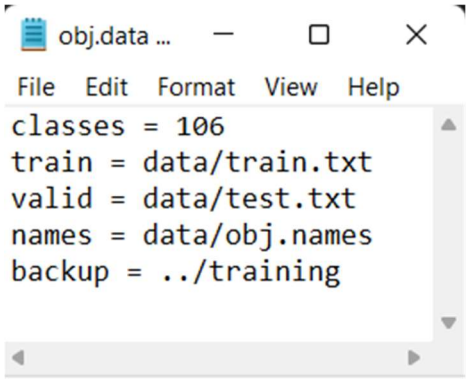
$$Steps_2 = Max\ batches\ .\frac{90}{100} \tag{3.3}$$

d. *Filter* pada setiap "*convolutional layer"* juga diganti mengikuti persamaan (3.4).

$$Filter = (class + 5)\,.\,3 \tag{3.4}$$

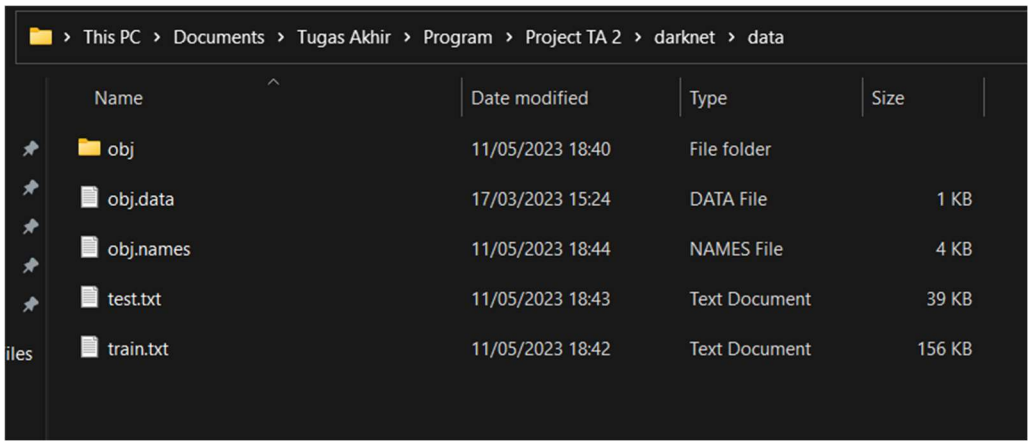
e. *Classes* pada setiap "*yolo layer*" akan diubah sesuai dengan jumlah kelas yang dipakai, yakni 106 *class*.

Setelah proses konfigurasi selesai, akan dibuat dua *file* pada *directory* "../darknet/data/" yaitu "obj.names" dan "obj.data". *file* "obj.names" merupakan *file* yang berisikan nama seluruh *class*, sedangkan "obj.data" berisikan metadata berupa banyaknya *class*, *path* dari *file* "*train*.txt" dan "*test*.txt", *path* dari *file* "obj.names", dan *backup* sebagai lokasi penyimpanan bobot. Gambar 3.13 merupakan isi dari *file* "obj.data".



Gambar 3.13 Isi *file* "obj.data"

Setelah *file* "obj.names" dan "obj.data" selesai dibuat, maka proses persiapan untuk pelatihan model arsitektur YOLOv4-*tiny* telah selesai dilakukan. Hasil persiapan dapat dilihat pada Gambar 3.14.



Gambar 3.14 Hasil dari persiapan pelatihan model YOLOv4-*tiny*

Pada Gambar 3.14 terdapat empat *file* dan satu *folder*. *File* "obj.data" merupakan *file* yang menjembatani arsitektur dengan *file* yang dibutuhkan, hal tersebut dikarenakan pada *file* "obj.data" terdapat *path* yang menghubungkan *folder* dan *file* lainnya.

Pelatihan pada YOLOv4-*tiny* menerapkan konsep *transfer learning*, maka dari itu perlu dilakukan pengunduhan bobot *pretrained* dari model tersebut, yaitu *"*yolov4-tiny.conv.29". Bobot *pretrained* YOLOv4-*tiny* ini telah dilatih sebelumnya hingga *29 convolutional layers.* Setelah diunduh, bobot *pretrained* YOLOv4-*tiny* diletakkan pada *directory* "../darknet/" agar dapat digunakan untuk pelatihan.

Sebelum memulai pelatihan arahkan terlebih *Command Prompt* (CMD) ke *directory* "../darknet/". Setelah itu, masukkan kode pada Gambar 3.15 untuk memulai pelatihan.

```
darknet.exe detector train data/obj.data cfg/yolov4-tiny-custom.cfg yolov4-tiny.conv.29 -dont_show -map
```

Gambar 3.15 Kode untuk memulai pelatihan

Pada Gambar 3.15 terdapat perintah untuk membuka program "darknet.exe". Kemudian program "darknet.exe" memanggil fungsi "*detector train*" untuk melakukan pelatihan dengan melihat *path* yang ada pada *file* "obj.data" untuk menemukan data *train*. Konfigurasi yang diterapkan berada pada *file* "yolov4-tiny-custom.cfg" dan menggunakan "yolov4-tiny.conv.29" sebagai bobot *pretrained*-nya. Selain itu terdapat parameter "*dont_show*" yang mengisyaratkan Darknet untuk tidak menampilkan *chart*, sedangkan parameter "map" merupakan opsi untuk menghitung *Mean Average Precision* (mAP).

Pada arsitektur YOLO khususnya YOLOv4-*tiny*, jika proses pelatihan berhenti secara tiba-tiba, pelatihan masih dapat dilanjutkan tanpa perlu mengulanginya dari awal. Proses untuk melanjutkan pelatihan dapat dengan memasukkan kode pada Gambar 3.16.

```
darknet.exe detector train data/obj.data cfg/yolov4-tiny-custom.cfg ../training/yolov4-tiny-custom_last.weights -dont_show -map
```

Gambar 3.16 Kode untuk melanjutkan pelatihan

Hal tersebut dapat terjadi karena YOLO akan melakukan penyimpanan bobot terakhir pada *file* "yolov4-tiny-custom_last.weights" di *folder training* secara otomatis setiap 100 iterasi. Selain itu, pada YOLOv4 juga menyediakan fungsi untuk menguji model yang telah dilatih. Fungsi tersebut dapat dilihat pada Gambar 3.17.

```
darknet.exe detector test data/obj.data cfg/yolov4-tiny-custom.cfg ../training/yolov4-tiny-custom_best.weights
../../mask_test_images/image1.jpg -thresh 0.3
```

Gambar 3.17 Kode untuk menguji model

Sebelum dapat menggunakan kode pada Gambar 3.17 untuk menguji model, terdapat dua *hyperparameter* yang harus dikonfigurasi, yakni *batch* dan *subdivisions*. Kedua *hyperparameter* tersebut harus diubah menjadi bernilai satu kembali. Fungsi pada Gambar 3.17 dapat dipahami sebagai perintah untuk membuka program "darknet.exe". Program "darknet.exe" kemudian memanggil fungsi "*detector test*" untuk melakukan pengujian deteksi objek, dengan melihat *path* yang ada pada *file* "obj.data" untuk menemukan data pengujian. Pengujian didasarkan pada konfigurasi "yolov4-tiny-custom.cfg" dan menggunakan "yolov4-tiny-custom_best.weights" sebagai model yang akan diuji. Parameter berikutnya yaitu berupa *path* ke gambar yang akan diuji. Selain itu terdapat parameter "*thresh*" yang merujuk pada *threshold* yang mengisyaratkan pengaturan nilai ambang batas lebih dari 0,3.

3.7 *Evaluation*

Pada tahap ini peneliti melakukan analisis performa terhadap model yang telah dikembangkan. Proses ini dapat dilakukan ketika semua proses *train model* telah selesai dilakukan. Tujuan dari proses ini untuk menentukan model mana yang terbaik, kemudian model tersebut akan dilakukan *convert model*. Pada arsitektur YOLOv4-*tiny* hasil evaluasi model sudah dijabarkan secara lengkap. Hasil evaluasi yang meliputi perhitungan nilai *average precision* (AP) pada setiap *class* yang digunakan. Selain itu, terdapat juga perhitungan *true positive* (TP), *false positive* (FP), false negative (FN), *precision, recall, f1-score, average IoU,* dan juga *mean average precision (mAP)* secara keseluruhan pada model yang telah dilakukan *training*. Akan tetapi, nilai *mean average precision (mAP) y*ang akan dijadikan tolak ukur pada penelitian ini. Hal tersebut karena *mAP* merupakan *metrics* yang umum atau populer digunakan dalam mengevaluasi *model object detection* dan mAP ini menggambarkan hasil nilai rata-rata keakuratan dari suatu model dalam melakukan deteksi objek di semua *class*. Proses evaluasi dilakukan dengan membandingkan mAP yang dihasilkan oleh model. Setelah didapatkan model yang terbaik berdasarkan mAP, selanjutnya model tersebut akan dianalisis secara keseluruhan.

3.8 *Testing*

Pada tahap ini dilakukan pengujian untuk mengetahui apakah sistem dapat bekerja dengan baik dalam mendeteksi objek produk retail, dalam kasus ini susu bubuk. Pengujian dilakukan dengan instalasi sistem pada perangkat *smartphone*. Model yang telah dipilih perlu diubah formatnya terlebih dahulu dari ".weight" menjadi ".tflite" agar dapat digunakan. Pengujian berfokus pada kebenaran prediksi serta waktu inferensi yang dibutuhkan. Terdapat dua perangkat *smartphone* yang digunakan pada tahap ini, dengan spesifikasi pada Tabel 3.3.

Tabel 3.3 Spesifikasi *smartphone* yang digunakan pada proses *testing*

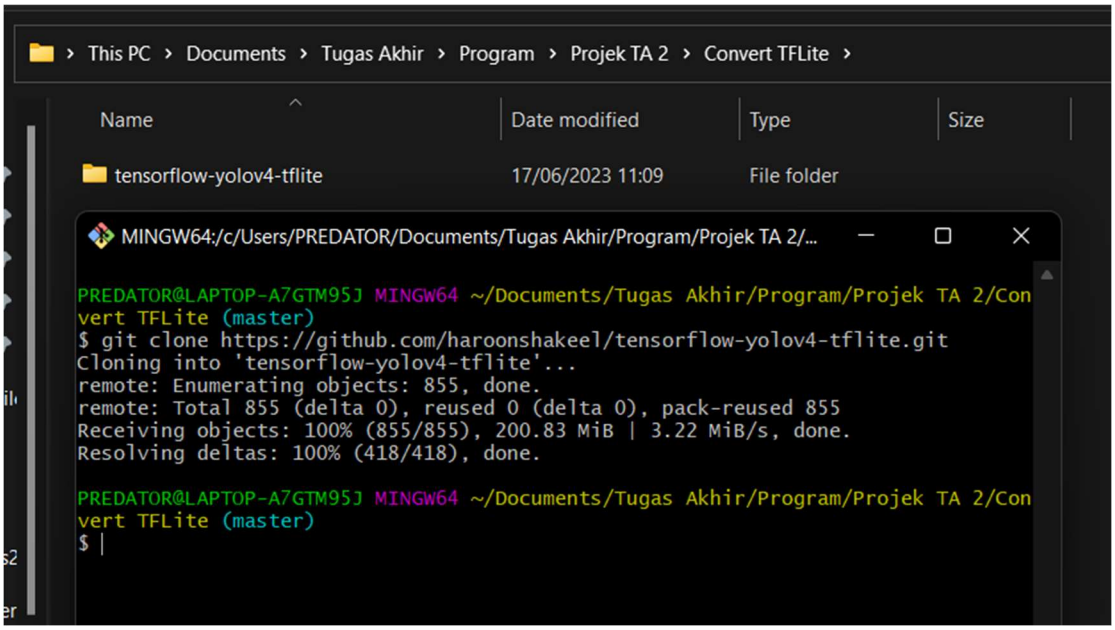
Komponen	Xiaomi Redmi Note 4x	Oppo A53
Processor	Snapdragon 625	Snapdragon 460
RAM	3 Gb	4 Gb
Resolusi Kamera	13 MP	13 MP

Pada Tabel 3.3 terdapat dua *smartphone* yang digunakan pada proses *testing model* di perangkat *mobile*. Perangkat pertama merupakan Xiaomi Redmi Note 4x merupakan produk smartphone yang keluar pada tahun 2017, dilengkapi dengan *processor* snapdragon 625, RAM 3 Gb, dan kamera tunggal dengan resolusi 13 MP. Perangkat kedua merupakan Oppo A53 merupakan produk smartphone yang keluar pada tahun 2020, dilengkapi dengan *processor* snapdragon 460, RAM 4 Gb, dan memiliki tiga kamera dengan resolusi kamera utama 13 MP. Berdasarkan pemaparan sebelumnya, Oppo A53 memiliki keunggulan pada sektor kamera, hal ini cukup penting untuk diketahui karena pengujian pada perangkat *mobile* berkaitan langsung dengan kamera yang digunakan.

3.8.1 *Convert Model*

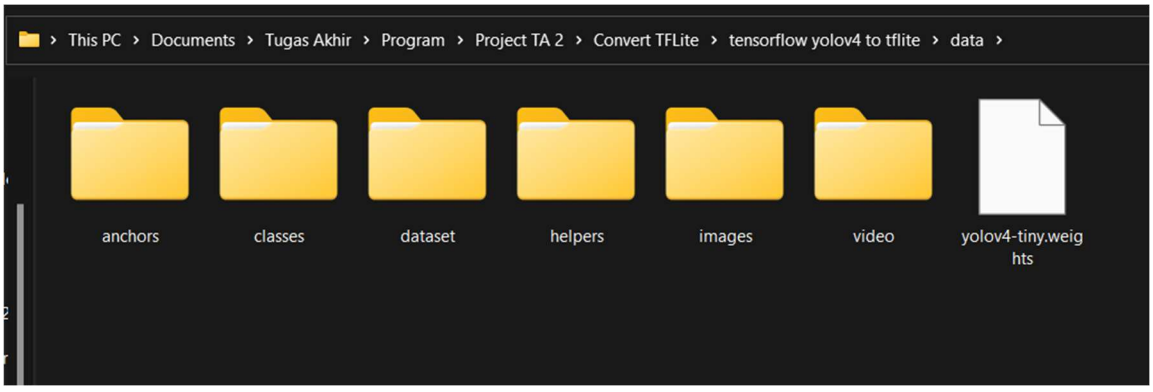
Sebelum model dapat dijalankan pada perangkat *smartphone* khususnya android, model perlu terlebih dahulu dilakukan *convert* model. Pada tahap ini peneliti menerapkan *project* yang telah dikembangkan oleh Haroon Shakeel (2020). Proses *convert* model tidak bisa langsung dari format ".*weight*" menjadi ".tflite" tapi perlu diubah terlebih dahulu ke format ".pb".

Hal pertama yang perlu untuk dilakukan yaitu melakukan proses *clone* terhadap *repository "tensorflow-yolov4-tflite"*. Proses *clone* ditunjukkan pada Gambar 3.18.



Gambar 3.18 *Clone repository* "*tensorflow-yolov4-tflite*"

Setelah proses *clone* berhasil dilakukan, *copy* model yang akan dilakukan *convert* ke dalam *directory* "../tensorflow yolov4 to tflite/data/", seperti pada Gambar 3.19. Kemudian *copy file* "obj.names" yang sebelumnya digunakan pada proses *train model* ke dalam *directory* "../tensorflow yolov4 to tflite/data/classes".



Gambar 3.19 *Copy* model yang akan dilakukan *convert* ke TFLite

Selanjutnya, buka *Anaconda Command* lalu arahkan ke *directory* "../tensorflow yolov4 to tflite/". Setelah itu, masukkan kode pada Gambar 3.20 untuk melakukan *convert* model.

```
python save_model.py --weights ./data/yolov4-tiny.weights --output ./checkpoints/yolov4-416 --input_size 416 --model yolov4 --framework tflite --tiny
```

Gambar 3.20 *Convert* bobot menjadi format ".pb"

Kode pada Gambar 3.20 dapat diartikan dengan perintah jalankan *file* "save_model.py" dengan parameter pertama merupakan lokasi *directory* dari *file* bobot yang akan dikonversi, parameter kedua merupakan lokasi model jika telah dilakukan *convert*, parameter ketiga merupakan ukuran *input size* yang diterima, parameter keempat merupakan *framework* yang digunakan untuk *convert* model, dan parameter terakhir merupakan tanda model yang dilakukan konversi merupakan versi *tiny*. Proses tersebut merupakan proses *convert* dari format ".weight" ke format ".pb". Gambar 3.21 merupakan kode untuk mengubah format dari ".pb" menjadi format TFLite.

```
python convert_tflite.py --weights ./checkpoints/yolov4-416 --output ./checkpoints/yolov4-416.tflite
```

Gambar 3.21 *Convert* ".pb" menjadi format ".tflite"

Kode pada Gambar 3.21 dapat diartikan dengan perintah jalankan *file* "convert_tflite.py" dengan parameter pertama merupakan lokasi *directory* dari *file* bobot yang akan dikonversi, parameter kedua merupakan lokasi model jika telah dilakukan *convert*. Gambar 3.22 merupakan kode untuk mengubah format dari ".pb" menjadi format TFLite dengan menerapkan kuantisasi.

```
python convert_tflite.py --weights ./checkpoints/yolov4-416 --output ./checkpoints/yolov4-416-fp16.tflite --quantize_mode float16
```

Gambar 3.22 *Convert* ".pb" menjadi format ".tflite" dengan menerapkan kuantisasi

Kode pada Gambar 3.22 sama seperti kode pada Gambar 3.21 hanya saja menerapkan kuantisasi "float16". Dikutip dari *website* resmi Tensorflow, penerapan kuantisasi "float16" pada format TFLite ini memiliki berbagai keuntungan antara lain:

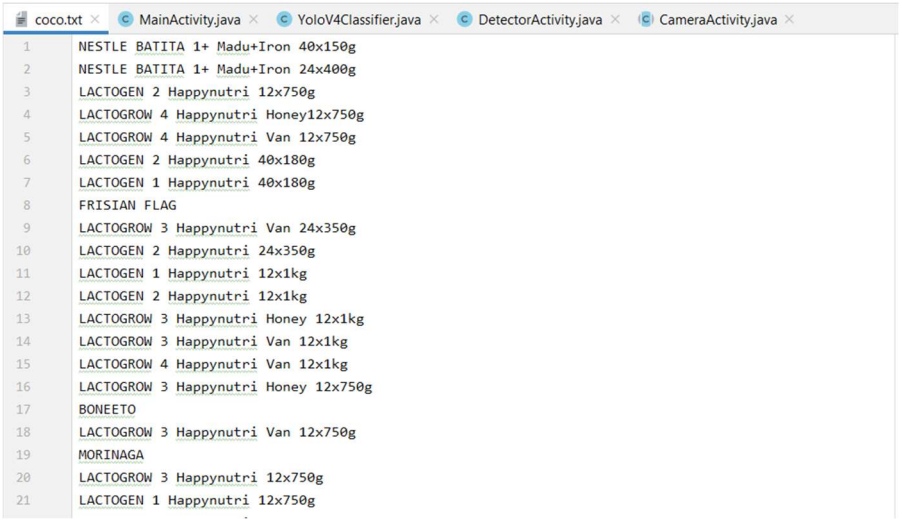
a. Ukuran model menjadi setengah dari ukuran aslinya.
b. Mengurangi terjadinya *loss* pada akurasi.
c. Hasil eksekusi yang lebih cepat.

3.8.2 *Menjalankan Model pada Perangkat Mobile*

Pada tahap ini model yang telah dilakukan *convert* akan diterapkan pada perangkat *smartphone* khususnya android. Penerapan dilakukan dengan menggunakan *Integrated Development Environment* (IDE) resmi untuk pengembangan perangkat lunak berbasis android. Pada tahap ini peneliti masih menerapkan *project* yang telah dikembangkan oleh Haroon Shakeel (2020). *Project* tersebut dipilih karena metode yang digunakan sama-sama mengenai pengembangan deteksi objek secara *real-time*.

Agar model dapat dijalankan pada *project* tersebut, diperlukan beberapa konfigurasi. Konfigurasi ini berkaitan dengan label serta model yang digunakan. Berikut merupakan langkah-langkah yang peneliti lakukan untuk menerapkan model pada perangkat *mobile* android:

a. Pertama, menyalin semua *class* label yang digunakan pada penelitian, yakni berjumlah 106 *class*. Kemudian data tersebut ditempelkan pada *file* "coco.txt" yang terletak pada *directory* "../tensorflow yolov4 to tflite/android/app/src/main/assets/". Gambar 3.23 merupakan hasil dari langkah pertama.



Gambar 3.23 Penyalinan *class label* ke *file* "coco.txt"

b. Kedua, menyalin model TFLite ke *directory* "../tensorflow yolov4 to tflite/android/app/src/main/assets/". Proses ini bertujuan agar model dapat terbaca pada *project*.

c. Ketiga, pada *file* "MainActivity.java" dan "DetectorActivity.java" terdapat variabel bernama "TF_OD_API_MODEL_FILE", variabel ini merujuk pada model yang akan

digunakan pada aplikasi. Isi dari variabel tersebut perlu diubah agar merujuk pada model penelitian ini.

d. Keempat, pada *file* "YoloV4Classifier.java" terdapat variabel bernama "isTiny", variabel ini merujuk pada tipe model yang akan digunakan pada aplikasi. Dikarenakan pada penelitian ini menggunakan arsitektur YOLOv4-*tiny* maka isi variabel tersebut diubah menjadi "*true*".

e. Kelima, setelah *project* dilakukan konfigurasi langkah berikutnya yaitu *build project*. Proses ini dilakukan untuk membuat *project* menjadi suatu aplikasi yang dapat dijalankan pada perangkat *mobile* android.

3.9 Perangkat Penelitian

Perangkat penelitian yang digunakan pada penelitian ini merupakan perangkat pribadi milik peneliti. Perangkat keras atau *hardware* merupakan faktor utama yang diperlukan dalam melakukan penelitian. Hal tersebut dikarenakan *hardware* sebagai otak untuk menjalankan pengolahan. *Hardware* yang mumpuni akan berpengaruh pada cepat atau lambatnya proses pengolahan, baik pengolahan *dataset* maupun *training model*. Peneliti menyelesaikan penelitian dengan menggunakan spesifikasi *hardware* seperti yang ada pada Tabel 3.4.

Tabel 3.4 Spesifikasi *hardware* yang digunakan dalam penelitian

Komponen	Spesifikasi
Processor	Intel(R) Core(TM) i5-9300H @ 2.40Ghz
RAM	8 Gb
GPU	NVIDIA GeForce GTX 1650 4Gb

Pada Tabel 3.4 perangkat penelitian menggunakan spesifikasi *hardware* dengan *processor* 4 *core* dan 8 *threads*, didukung NVIDIA GeForce GTX 1650 yang memiliki *compute capability CUDA* sebesar 7.5. Selain itu, perangkat ini juga dilengkapi dengan RAM 8 Gb untuk menunjang performanya. Komponen ini merupakan komponen yang cukup untuk melakukan *training* model YOLOv4-*tiny*.

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan mengenai hasil dari berbagai proses penerapan sistem deteksi objek. Hasil-hasil yang dibahas meliputi hasil pengolahan *dataset*, hasil pelatihan dari model deteksi objek, serta hasil dari penerapan YOLOv4-*tiny* pada perangkat *mobile*.

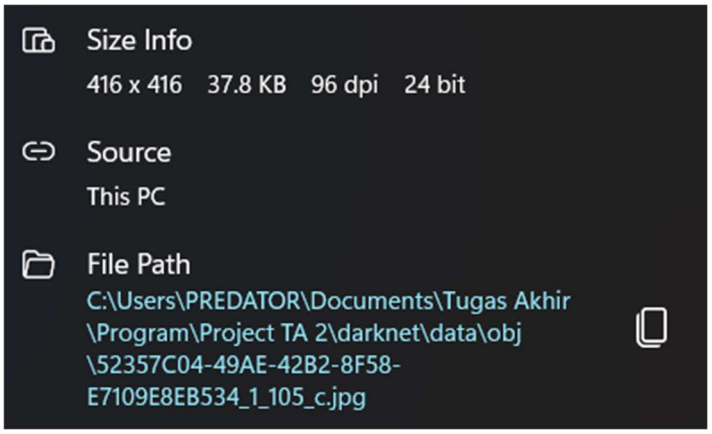
4.1 Hasil Pengolahan *Dataset*

Pada subbab ini menjelaskan mengenai hasil dari proses *preprocessing* data dan juga hasil pelabelan. Data yang dihasilkan dari proses ini, akan digunakan sebagai *dataset* dalam proses *training model*.

4.1.1 Hasil Proses *Preprocessing*

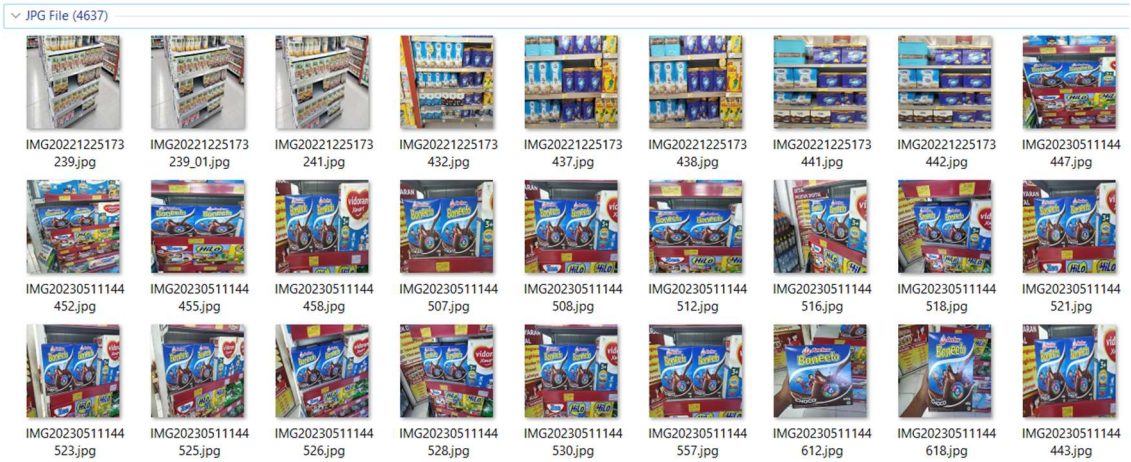
Preprocessing yang dilakukan pada penelitian ini meliputi pengaturan resolusi, *sharpening*, dan pengubahan format. Proses ini merupakan proses penentu bagus atau tidaknya suatu *dataset*. Adapun contoh hasil *preprocessing* pada penelitian ini seperti pada Gambar 4.1, sedangkan pada Gambar 4.2 merupakan contoh metadata *dataset* yang telah dilakukan *preprocessing*.

Gambar 4.1 Contoh data gambar yang telah dilakukan *preprocessing*



Gambar 4.2 Metadata suatu *dataset* setelah dilakukan *preprocessing*

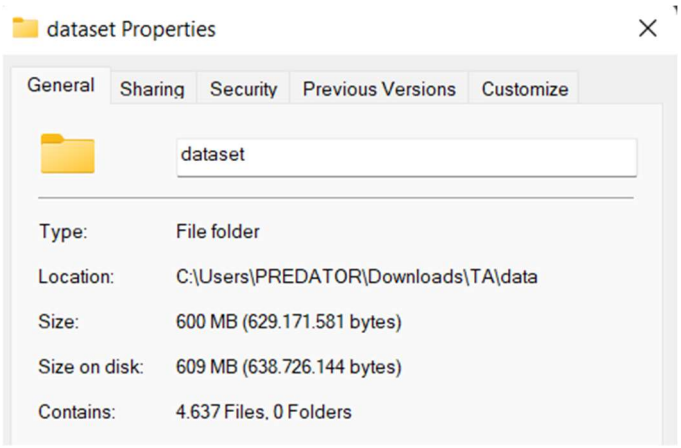
Pada Gambar 4.1 dan Gambar 4.2 terlihat bahwa data gambar yang sebelumnya memiliki berbagai resolusi telah diubah menjadi resolusi 416x416 *pixels*. Pengubahan ini tentunya memberikan dampak yang besar bagi gambar diantaranya ukuran objek menjadi berbeda dari ukuran aslinya. Pengubahan resolusi ini akan mengurangi ukuran pada data, yang menjadikan ukurannya lebih kecil. Selain dilakukan pengaturan resolusi, data gambar juga dilakukan *sharpening* dan data gambar disimpan dengan format gambar ".jpg". Hal ini menjadikan gambar semakin tajam sehingga batas antar objek semakin jelas. Pada Gambar 4.3 menunjukkan hasil dari proses *preprocessing.*



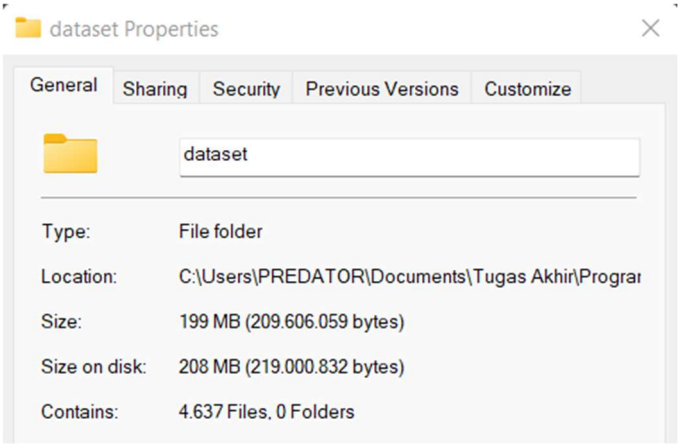
Gambar 4.3 Hasil data gambar yang telah dilakukan *preprocessing*

Pada Gambar 4.3 terlihat bahwa sebanyak 4637 data gambar telah memiliki format serta resolusi yang sama dikumpulkan ke dalam satu *folder*. Data-data inilah yang digunakan pada proses *training*. Hasil dari proses *preprocessing* ini memberikan dampak yang signifikan untuk

ukuran data. Ukuran data sebelum dilakukan proses *preprocessing* ditunjukkan pada Gambar 4.4 dan ukuran data setelah dilakukan proses *preprocessing* ditunjukkan pada Gambar 4.5.



Gambar 4.4 Ukuran data gambar sebelum *preprocessing*



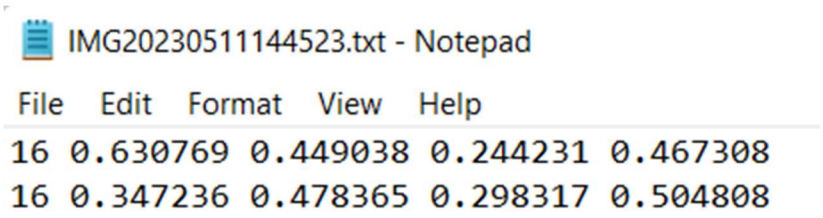
Gambar 4.5 ukuran data gambar sesudah *preprocessing*

Pada Gambar 4.5 terlihat bahwa setelah dilakukan *preprocessing* ukuran seluruh data gambar berubah. Sebelum dilakukan *preprocessing* seluruh data berukuran 600 MB, setelah dilakukan proses ini data gambar menjadi berukuran 200 MB. Dengan ukuran data yang lebih kecil akan mengurangi waktu yang diperlukan pada saat *training*.

4.1.2 Hasil Proses *Annotation*

Sebanyak 4637 data gambar yang telah diberi label dengan format data YOLO, akan disimpan dalam format ".txt" pada *folder* yang sama. Setiap satu *file* ".txt" berisikan semua *object* pada gambar yang telah diberi label dengan isian terdiri dari *id class*, koordinat titik

tengah sumbu x dan y dari objek, serta lebar dan tinggi dari objek. Hasil dari proses pelabelan ditunjukkan pada Gambar 4.6.



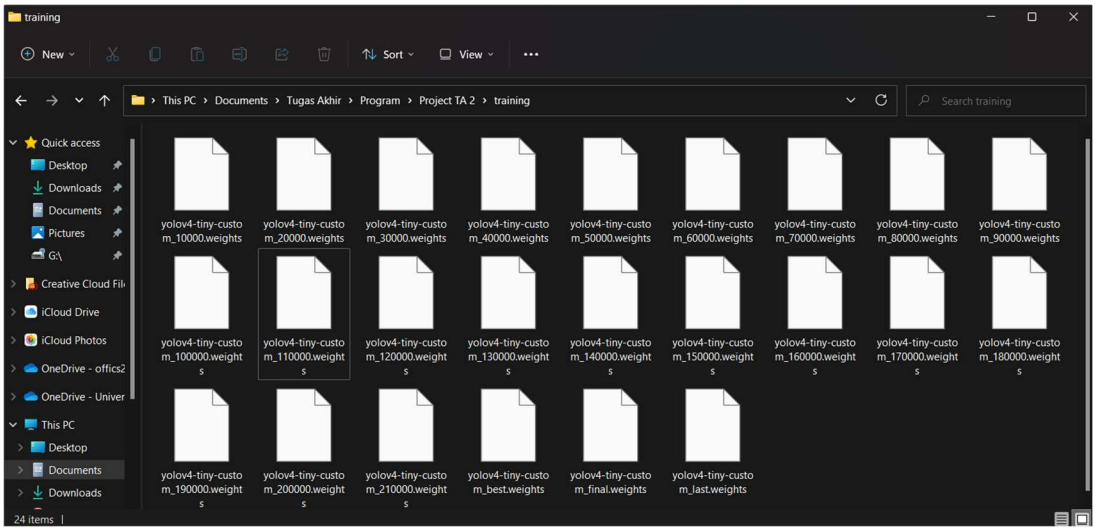
Gambar 4.6 Hasil *Image Annotation*

Gambar 4.6 merupakan contoh hasil dari proses *image annotation*. Isi dari *file* ".txt" terdiri atas:

a. Kolom pertama merupakan class_id objek. Class_id 16 merepresentasikan objek dengan label "BONNETO".
b. Kolom kedua dan ketiga mewakili titik *center* dari sumbu x dan titik *center* dari sumbu
c. Kolom keempat dan kelima melambangkan lebar dan tinggi dari kotak pembatas label.

4.2 Hasil *Training Model*

Proses *training* yang dilakukan dengan arsitektur YOLOv4-*tiny* menghasilkan *output* berupa *chart* dan model YOLOv4-*tiny* dengan *file* ekstensi "*.weight*". Selama proses *training* dilakukan, model akan menyimpan ke dalam tiga pembobotan seperti pada Gambar 4.7.



Gambar 4.7 Penyimpanan hasil proses *training*

Pada Gambar 4.7 dapat dilihat ketiga pembobotan yang terdiri atas:

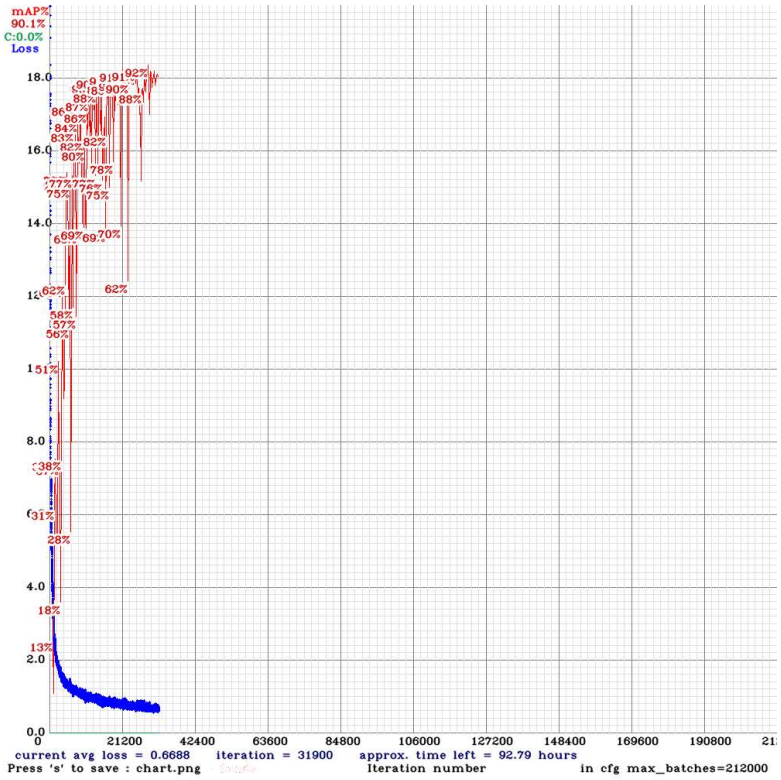
a. Pembobotan setiap 10.000 *steps* merupakan pembobotan yang disimpan *training* telah mencapai kelipatan 10.000 *steps*.

b. Pembobotan "*best.weight*" merupakan pembobotan terbaik selama melakukan proses *training*.

c. Pembobotan "*last.weight*" merupakan *steps* terakhir yang dilakukan pada saat *training* dan disimpan setiap 100 *steps*.

Proses pelatihan ini memakan waktu sekitar yang cukup lama, yakni 111 jam. Hal tersebut berbanding lurus dengan banyaknya kelas yang digunakan, yakni 106 kelas dengan "*max_batches"* sebanyak 212.000 *steps*. Saat melakukan pelatihan, peneliti menyiasati dengan membagi sesi *training* ke dalam enam sesi yang menghabiskan waktu lima hari. Hal ini berdampak pada terpisahnya *output chart* dan "*best.weight*".

Pada sesi pertama *training* ini memakan waktu 27 jam dan menghasilkan *output chart* seperti pada Gambar 4.8. Selain itu, *output* "*best.weight*" pada sesi pertama ini ditunjukkan pada Gambar 4.9.



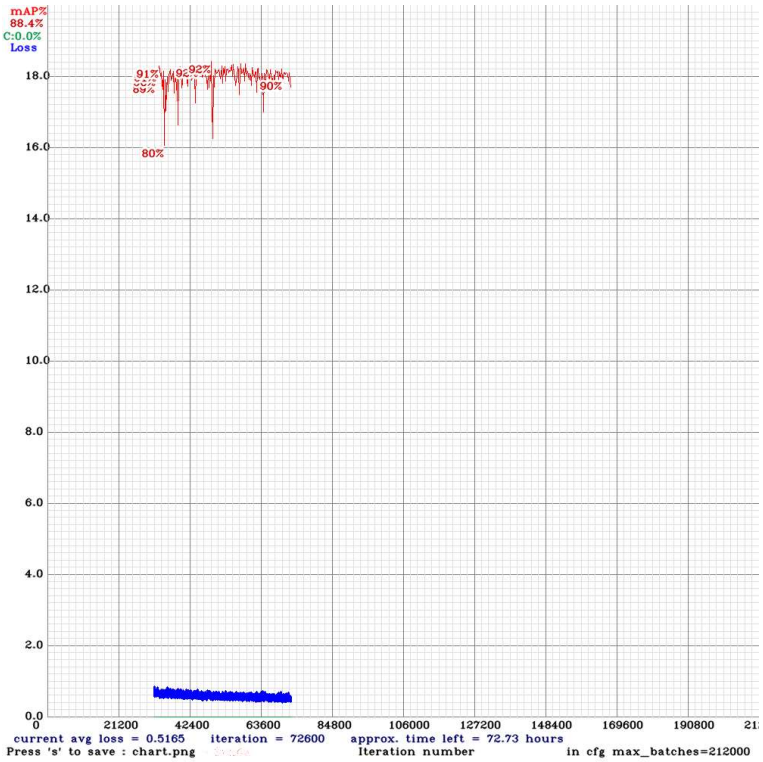
Gambar 4.8 *Chart* Sesi Pertama Proses *Training*

```
 for conf_thresh = 0.25, precision = 0.78, recall = 0.85, F1-score = 0.81
 for conf_thresh = 0.25, TP = 6053, FP = 1719, FN = 1067, average IoU = 63.37 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.916803, or 91.68 %
Total Detection Time: 21 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet>
```

Gambar 4.9 Sesi Pertama Proses *Training*

Berdasarkan kedua gambar tersebut sesi pertama ini didapatkan mAP terbaik sebesar 91,68%. Pada *chart* diketahui bahwa pada sesi pertama ini nilai mAP yang dihasilkan oleh model masih belum stabil. Selain itu, iterasi terakhir yang dilakukan pada sesi pertama ini yaitu 31.900 *steps*.

Pada sesi kedua *training* yang memakan waktu 19 jam yang menghasilkan *output chart* seperti pada Gambar 4.10. Untuk hasil *output* "*best.weight*" pada sesi kedua ini ditunjukkan pada Gambar 4.11.



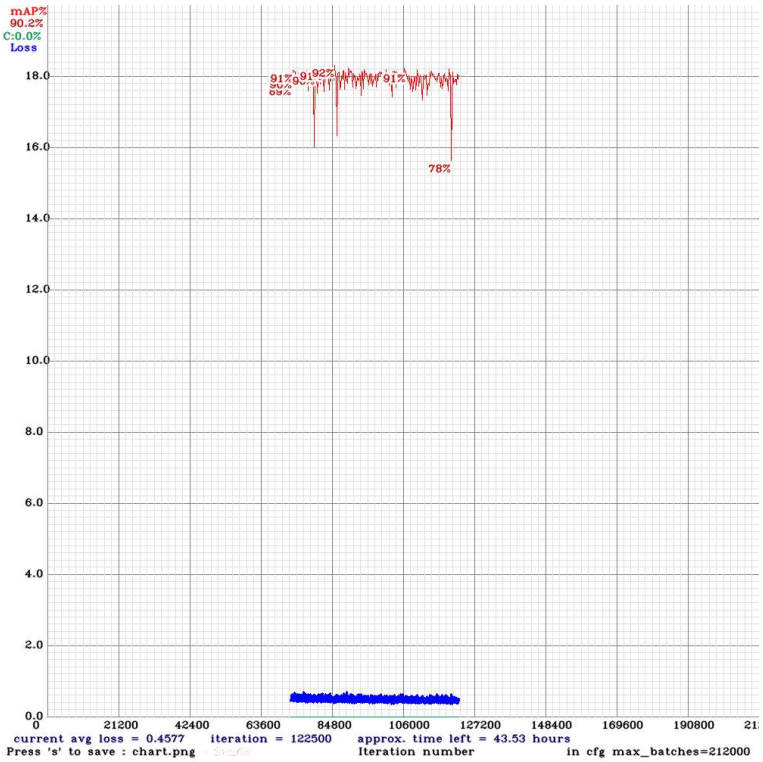
Gambar 4.10 *Chart* Sesi Kedua Proses *Training*

```
 for conf_thresh = 0.25, precision = 0.78, recall = 0.82, F1-score = 0.80
 for conf_thresh = 0.25, TP = 5832, FP = 1634, FN = 1288, average IoU = 62.55 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.921176, or 92.12 %
Total Detection Time: 20 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet>
```

Gambar 4.11 Sesi Kedua Proses *Training*

Berikutnya pada sesi kedua ini didapatkan mAP terbaik sebesar 92,12%. Pada *chart* diketahui bahwa pada sesi kedua ini nilai mAP yang dihasilkan oleh model sudah mulai stabil. Selain itu, iterasi terakhir yang dilakukan pada sesi kedua ini yaitu 72.600 *steps*.

Pada sesi ketiga *training* yang memakan waktu 23 jam yang menghasilkan *output chart* seperti pada Gambar 4.12. Untuk hasil *output* "*best.weight*" pada sesi ketiga ini ditunjukkan pada Gambar 4.13.



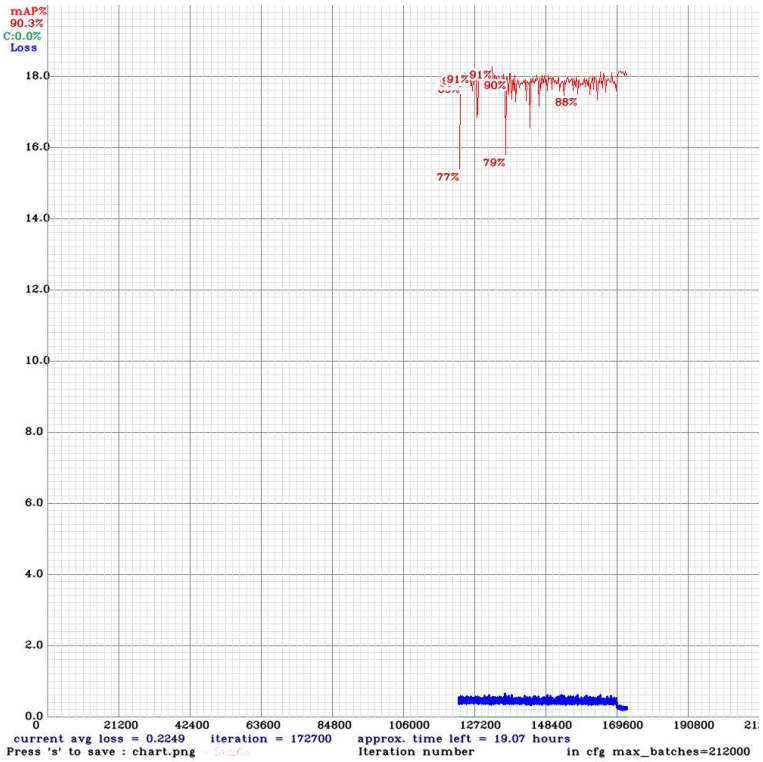
Gambar 4.12 *Chart* Sesi Ketiga Proses *Training*

```
 for conf_thresh = 0.25, precision = 0.78, recall = 0.82, F1-score = 0.80
 for conf_thresh = 0.25, TP = 5853, FP = 1649, FN = 1267, average IoU = 65.28 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.914549, or 91.45 %
Total Detection Time: 18 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet>
```

Gambar 4.13 Sesi Ketiga Proses *Training*

Berikutnya pada sesi ketiga ini didapatkan mAP terbaik sebesar 91,45%. Pada *chart* diketahui bahwa pada sesi ketiga ini nilai mAP yang dihasilkan oleh model masih sama seperti sesi kedua. Selain itu, iterasi terakhir yang dilakukan pada sesi ketiga ini yaitu 122.500 *steps*.

Pada sesi keempat *training* yang memakan waktu 23 jam yang menghasilkan *output chart* seperti pada Gambar 4.14. Untuk hasil *output* "*best.weight*" pada sesi keempat ini ditunjukkan pada Gambar 4.15.



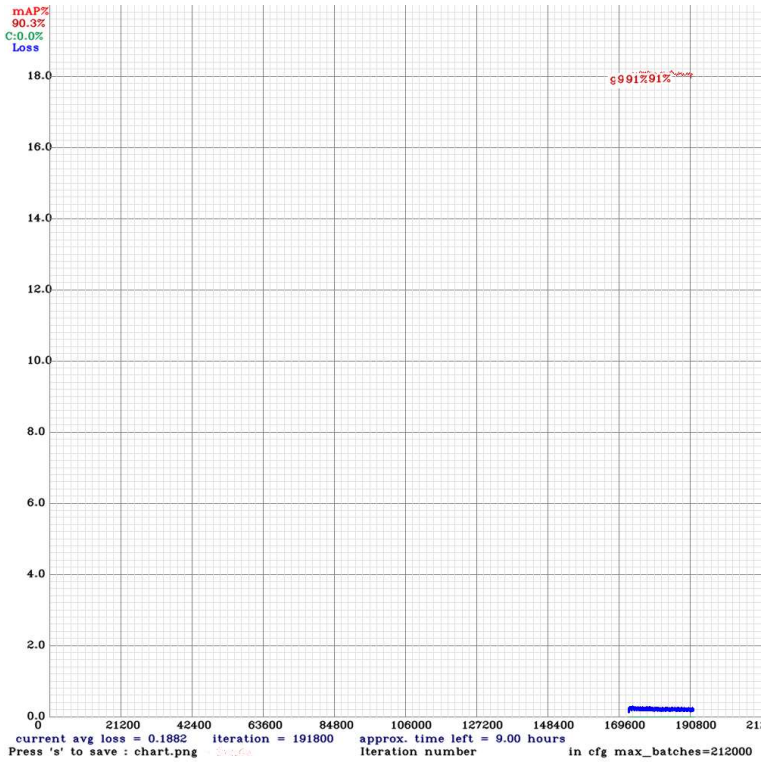
Gambar 4.14 *Chart* Sesi Keempat Proses *Training*

```
 for conf_thresh = 0.25, precision = 0.79, recall = 0.85, F1-score = 0.82
 for conf_thresh = 0.25, TP = 6034, FP = 1638, FN = 1086, average IoU = 64.39 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.913020, or 91.30 %
Total Detection Time: 18 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet>
```

Gambar 4.15. Sesi Keempat Proses *Training*

Berikutnya pada sesi keempat ini didapatkan mAP terbaik sebesar 91,30%. Pada *chart* diketahui bahwa pada sesi keempat ini nilai mAP yang dihasilkan oleh model masih sama seperti sebelumnya. Selain itu, iterasi terakhir yang dilakukan pada sesi keempat ini yaitu 172.700 *steps*.

Pada sesi kelima *training* yang memakan waktu 9 jam yang menghasilkan *output chart* seperti pada Gambar 4.16. Untuk hasil *output* "*best.weight*" pada sesi kelima ini ditunjukkan pada Gambar 4.17.



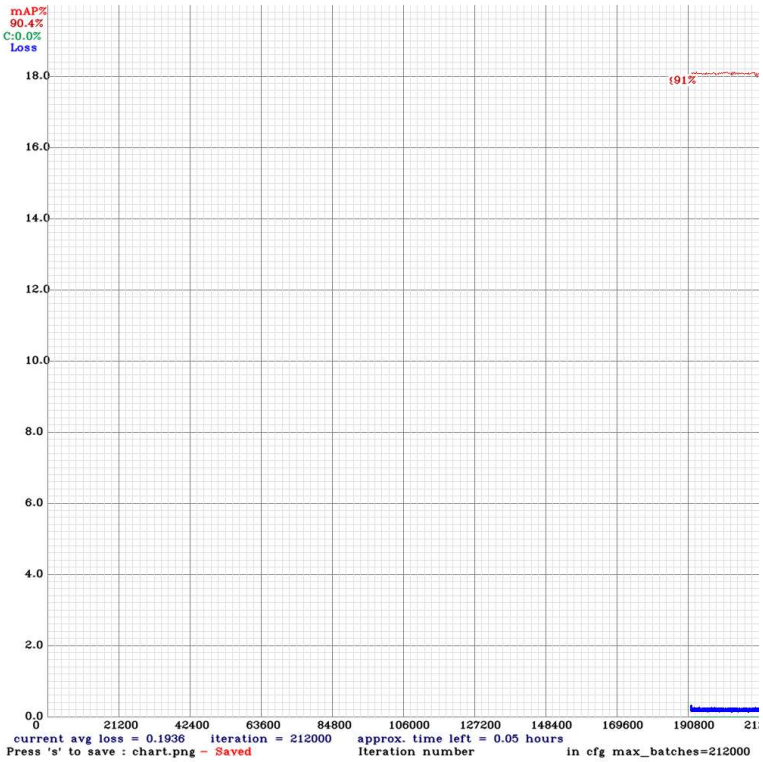
Gambar 4.16 *Chart* Sesi Kelima Proses *Training*

```
 for conf_thresh = 0.25, precision = 0.79, recall = 0.80, F1-score = 0.80
 for conf_thresh = 0.25, TP = 5728, FP = 1498, FN = 1392, average IoU = 71.82 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.907857, or 90.79 %
Total Detection Time: 19 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset

C:\Users\PREDATOR\Documents\Tugas Akhir\Program\Project TA 2\darknet>
```

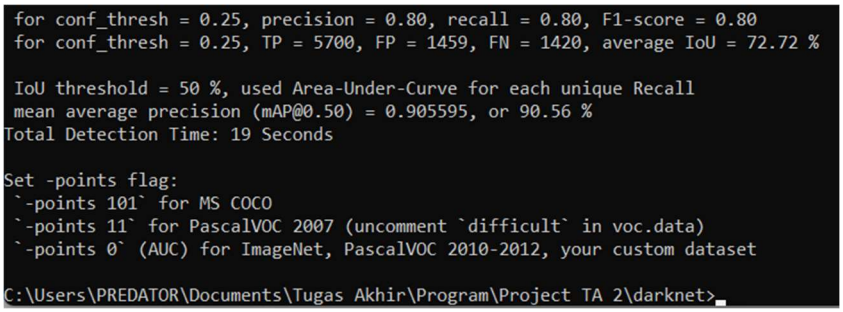
Gambar 4.17 Sesi Kelima Proses *Training*

Berikutnya pada sesi kelima ini didapatkan mAP terbaik sebesar 90,79%. Pada *chart* diketahui bahwa pada sesi kelima ini nilai mAP yang dihasilkan oleh model sudah stabil. Selain itu, iterasi terakhir yang dilakukan pada sesi kelima ini yaitu 191.800 *steps*.

Pada sesi keenam *training* yang memakan waktu 10 jam yang menghasilkan *output chart* seperti pada Gambar 4.18. Untuk hasil *output* "*best.weight*" pada sesi keenam ini ditunjukkan pada Gambar 4.19.



Gambar 4.18 *Chart* Sesi Keenam Proses *Training*



Gambar 4.19 Sesi Keenam Proses *Training*

Berikutnya pada sesi keenam ini didapatkan mAP terbaik sebesar 90,56%. Pada *chart* diketahui bahwa pada sesi keenam ini nilai mAP yang dihasilkan oleh model sudah mulai stabil. Selain itu, iterasi terakhir yang dilakukan pada sesi keenam ini yaitu 212.000 *steps* yang mana merupakan iterasi terakhir dari proses pelatihan.

Setelah seluruh sesi *training* selesai dijalankan maka akan terlihat sesi mana yang memiliki mAP tertinggi. Rincian mAP dari setiap sesi pada proses *training* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil mAP dari proses *training* setiap sesi

Weight	mAP
Best1	91,68%
Best2	92,12%
Best3	91,45%
Best4	91,30%
Best5	90,79%
Best6	90,56%

Dari keenam sesi *training,* diketahui bahwa sesi *training* kedua merupakan sesi terbaik karena berhasil mendapatkan mAP yang tertinggi daripada sesi yang lainnya, yakni sebesar 92,12%. Secara lebih detail hasil *evaluation* pada sesi kedua ditunjukkan pada Tabel 4.2.

Tabel 4.2 Rincian *Evaluation training* terbaik

Evaluation	**Nilai**
TP	5833
FP	1633
FN	1287
Precision	0,78
Recall	0,82
F1-score	0,80
Average IoU	62,56%

Evaluation	Nilai
mAP	92,12%

Berdasarkan hasil *evaluation* pada Tabel 4.2 apabila dianalisis nilai yang dihasilkan dari bobot tersebut sudah cukup baik untuk digunakan. Oleh karena itu, hasil pelatihan pada sesi kedua (*best2*) dipilih sebagai *weight* YOLOv4-*tiny* pada penelitian ini. Hasil *Evaluation* setiap label dari proses *training* ditunjukkan pada Tabel 4.3.

Tabel 4.3 Hasil *Evaluation* Setiap Label

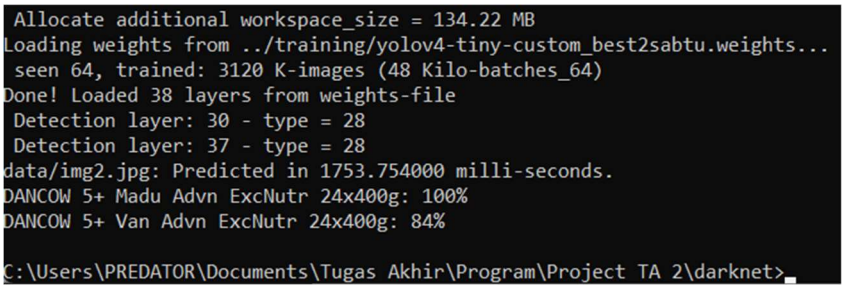
Class_id	Label	AP
0	NESTLE BATITA 1+ Madu+Iron 40x150g	100,00%
1	NESTLE BATITA 1+ Madu+Iron 24x400g	97,78%
2	LACTOGEN 2 Happynutri 12x750g	96,41%
3	LACTOGROW 4 Happynutri Honey12x750g	94,07%
4	LACTOGROW 4 Happynutri Van 12x750g	94,70%
5	LACTOGEN 2 Happynutri 40x180g	96,18%
6	LACTOGEN 1 Happynutri 40x180g	93,87%
7	FRISIAN FLAG	89,00%
8	LACTOGROW 3 Happynutri Van 24x350g	81,54%
9	LACTOGEN 2 Happynutri 24x350g	93,60%
10	LACTOGEN 1 Happynutri 12x1kg	89,38%
11	LACTOGEN 2 Happynutri 12x1kg	98,08%
12	LACTOGROW 3 Happynutri Honey 12x1kg	92,91%
13	LACTOGROW 3 Happynutri Van 12x1kg	99,01%
14	LACTOGROW 4 Happynutri Van 12x1kg	83,15%
15	LACTOGROW 3 Happynutri Honey 12x750g	91,00%
16	BONEETO	80,00%
17	LACTOGROW 3 Happynutri Van 12x750g	84,72%
18	MORINAGA	67,33%
19	LACTOGROW 3 Happynutri 12x750g	92,28%
20	LACTOGEN 1 Happynutri 12x750g	95,28%
21	LACTOGEN 1 Happynutri 24x350g	94,81%
22	LACTOGROW 3 Happynutri Honey 24x350g	93,71%
23	DANCOW 5+ Van Advn ExcNutr 24x400g	90,66%
24	DANCOW 5+ Madu Advn ExcNutr 24x400g	97,93%
25	DANCOW 5+ Cok Advn ExcNutr 24x400g	94,03%
26	DANCOW 5+ Cok Advn ExcNutr 12x1000g	98,91%
27	DANCOW 3+ Van Advn ExcNutr 12x800g	93,62%
28	DANCOW 3+ Madu Advn ExcNutr 12x800g	94,51%
29	DANCOW 3+ Cok Advn ExcNutr 12x800g	98,28%
30	DANCOW 3+ Cok Advn ExcNutr 24x400g	98,13%
31	DANCOW 5+ Cok Advn ExcNutr 12x800g	94,27%
32	DANCOW 5+ Madu Advn ExcNutr 12x800g	79,51%
33	DANCOW 3+ Madu Advn ExcNutr 12x1000g	87,70%
34	DANCOW 1+ Madu Advn ExcNutr 12x800g	94,37%
35	DANCOW 5+ Van Advn ExcNutr 12x800g	91,58%
36	DANCOW 1+ Van Advn ExcNutr 12x800g	92,91%

Class_id	Label	AP
37	DANCOW 1+ Cok Advn ExcNutr 12x800g	98,68%
38	DANCOW 1+ Van Advn ExcNutr 24x400g	89,81%
39	DANCOW 1+ Madu Advn ExcNutr 24x400g	94,40%
40	DANCOW 3+ Madu Advn ExcNutr 40x200g	96,13%
41	NESTLE DATITA 3+ Madu+Iron 24x400g	96,81%
42	DANCOW 3+ Van Advn ExcNutr 24x400g	93,09%
43	DANCOW 3+ Madu Advn ExcNutr 24x400g	94,70%
44	NESTLE BATITA 1+Vanilla+Iron 12x900g	92,33%
45	NESTLE DATITA 3+ Madu+Iron 12x900g	88,76%
46	LACTOGROW 3 Happynutri Honey 40x180g	74,71%
47	DANCOW 1+ Cok Advn ExcNutr 24x400g	99,58%
48	S-26 PROCAL Pouch 12x400g	100,00%
49	S-26 PROMISE Pouch 12x400g	100,00%
50	S-26 PROCAL Pouch 12x700g	100,00%
51	S-26 PROMIL 1 Pouch 12x400g	100,00%
52	S-26 PROMIL 2 Pouch 12x400g	99,46%
53	S-26 PROMISE Pouch 12x700g	99,93%
54	S-26 PROCAL Pouch 6,2x700g	100,00%
55	LACTOGROW 3 Happynutri 24x350g	78,45%
56	DANCOW 5+ Van Advn ExcNutr 24x400g REDESIGN	93,35%
57	NESTLE BATITA 1+ Vanilla+Iron 24x400g	99,83%
58	NESTLE BATITA 1+ Madu+Iron 12x900g	93,70%
59	DANCOW 3+ Madu Advn ExcNutr 12x800g REDESIGN	96,09%
60	DANCOW 5+ Madu Advn ExcNutr 12x800g REDESIGN	97.37%
61	DANCOW 5+ Madu Advn ExcNutr 12x1000g	81,82%
62	NESTLE DATITA 3+ Madu+Iron 40x150g	100,00%
63	NESTLE DATITA 3+ Vanilla+Iron 24x400g	100,00%
64	SGM	34,67%
65	VIDORAN	78,10%
66	ENFAGROW	80,30%
67	NESTLE DATITA 3+Vanilla+Iron 12x900g	78,60%
68	NESTLE DATITA 5+ Madu+Iron 12x900g	100,00%
69	DANCOW 1+ Cok Advn ExcNutr 12x800g REDESIGN	100,00%
70	DANCOW 1+ Madu Advn ExcNutr 12x1000g	95,96%
71	HILO	28,88%
72	DANCOW 3+ Van Advn ExcNutr 12x1000g	50,00%
73	LACTOGROW 4 Happynutri Van 12x1kg REDESIGN	100,00%
74	LACTOGROW 4 Happynutri Honey12x1kg	90,97%
75	LACTOGROW 3 Happynutri Van 12x1kg REDESIGN	100,00%
76	DANCOW 5+ Cok Advn ExcNutr 12x800g REDESIGN	96,84%
77	DANCOW 5+ Van Advn ExcNutr 12x800g REDESIGN	97,84%
78	DANCOW 1+ Van Advn ExcNutr 40x200g REDESIGN	100,00%
79	DANCOW 3+ Van Advn ExcNutr 24x400g REDESIGN	100,00%
80	DANCOW 3+ Madu Advn ExcNutr 24x400g REDESIGN	100,00%
81	DANCOW 1+ Madu Advn ExcNutr 24x400g REDESIGN	100,00%
82	DANCOW 1+ Van Advn ExcNutr 24x400g REDESIGN	100,00%
83	DANCOW 5+ Madu Advn ExcNutr 24x400g REDESIGN	100,00%
84	DANCOW 1+ Madu Advn ExcNutr 12x1000g REDESIGN	100,00%
85	DANCOW 1+ Van Advn ExcNutr 12x1000g REDESIGN	99,24%
86	DANCOW 3+ Van Advn ExcNutr 12x1000g REDESIGN	100,00%

Class_id	Label	AP
87	DANCOW 3+ Madu Advn ExcNutr 12x1000g REDESIGN	100,00%
88	DANCOW 5+ Madu Advn ExcNutr 12x1000g REDESIGN	98,05%
89	DANCOW 1+ Van Advn ExcNutr 12x800g REDESIGN	85,28%
90	DANCOW 1+ Madu Advn ExcNutr 12x800g REDESIGN	91,78%
91	DANCOW 3+ Van Advn ExcNutr 12x800g REDESIGN	93,14%
92	DANCOW 3+ Cok Advn ExcNutr 12x800g REDESIGN	95,37%
93	DANCOW 3+ Madu Advn ExcNutr 40x200g REDESIGN	92,04%
94	DANCOW 1+ Madu Advn ExcNutr 40x200g REDESIGN	95,33%
95	DANCOW 1+ Van Advn ExcNutr 12x1000g	83,33%
96	DANCOW 1+ Van Advn ExcNutr 40x200g	90,74%
97	PROVITAL	78,61%
98	NUTRICIA	94,06%
99	DANCOW 1+ Madu Advn ExcNutr 40x200g	96,32%
100	S-26 GOLD PROMIL 2 Can Top 6x900g	92,14%
101	S-26 GOLD PROMIL 1 Can Top 6x900g	100,00%
102	S-26 PROMISE GOLD Can Top 6x900g	97,03%
103	S-26 PROMISE GOLD Can 6x1,6kg	98,88%
104	S-26 PROCAL GOLD Can 6x1,6kg	99,95%
105	S-26 PROCAL GOLD Can Top 6x900g	89,28%

Berdasarkan hasil pada Tabel 4.3 dapat diketahui bahwa sebagian besar *class* memiliki nilai *AP* yang sudah cukup baik, yakni mencapai lebih dari 80%. Hal tersebut menunjukkan bahwa model dapat melakukan deteksi dengan baik di sebagian besar *class* yang ada. Selain itu, terdapat pula beberapa *class* memiliki nilai *AP* yang sempurna mencapai 100%, hal ini dapat terjadi dikarenakan prediksi yang dilakukan oleh model tepat pada *class* tersebut. Terdapat pula *class* yang memiliki nilai *AP* yang rendah dengan nilai AP kurang dari 50%, seperti pada *class* 64 dan 71. Hal ini terjadi dikarenakan model tidak dapat dengan benar mendeteksi objek pada *class* tersebut. Pada Gambar 4.20 dan Gambar 4.21 merupakan uji coba pada model untuk mendeteksi gambar produk susu.

Gambar 4.20 Hasil Prediksi Model Pada Gambar



Gambar 4.21 Hasil Prediksi Model Pada *Command Prompt*

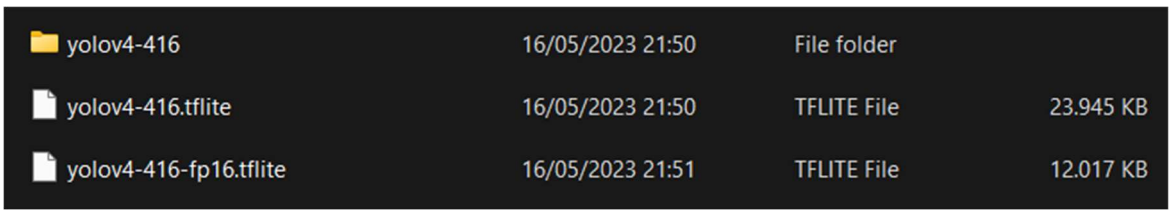
Pada kedua gambar tersebut dapat diketahui bahwa model YOLOv4-*tiny* mampu mendeteksi dua objek dengan benar dalam waktu 1753ms. Produk yang berhasil dideteksi yaitu "Dancow 5+ Madu Advanced Excelnutri 400g" dengan *threshold* 100% dan "Dancow 5+ Vanila Advanced Excelnutri 400g" dengan *threshold* 84%. Hal tersebut menunjukkan bahwa YOLOv4-*tiny* cukup akurat dalam mengenali objek produk retail susu bubuk yang ditampilkan dengan bantuan *bounding box,* walaupun resolusi gambar berkurang yang dikarenakan pengaturan resolusi.

4.3 Hasil Penerapan pada Perangkat *Mobile*

Pada subbab ini akan dijelaskan mengenai hasil dari pengubahan format model YOLO ke bentuk TFLite dan hasil penerapan pada perangkat *mobile*.

4.3.1 Hasil Pengubahan Bobot Ke Dalam Bentuk TFLite

Format model yang telah dilakukan konversi ke TFLite, akan berubah menjadi ".tflite". Hal tersebut dilakukan agar model dapat dijalankan pada perangkat *mobile*. Hasil proses ini ditunjukkan pada Gambar 4.22.



Gambar 4.22 Hasil *Convert* Format ".weight" ke ".tflite"

Pada Gambar 4.22 dapat dilihat bahwa proses pengubahan ini menghasilkan dua *file* dengan format TFLite. Pada *file* pertama yaitu "yolov4-416.tflite" merupakan hasil konversi langsung dari format YOLO, sedangkan pada *file* kedua yaitu "yolov4-416-fp16.tflite"

merupakan hasil konversi yang dilakukan kuantisasi float16. Jika dibandingkan dengan *file* pertama, pada *file* kedua ini memiliki ukuran yang jauh lebih kecil. Berdasarkan hal tersebut, pada penelitian ini digunakan hasil konversi ke tflite dengan kuantisasi "float16".

4.3.2 Hasil Pengujian Pada Perangkat *Mobile*

Penerapan pada perangkat *mobile* dilakukan untuk melihat bagaimana inferensi waktu yang dihasilkan oleh *model object detection*. Hal tersebut penting dilakukan untuk menunjukkan bahwa model dapat berjalan secara *real-time*. Uji coba menggunakan perangkat *mobile* ditunjukkan pada Tabel 4.4.

Tabel 4.4 Hasil uji coba pada perangkat *mobile*

Hasil Pengujian Pada Perangkat Xiaomi Redmi Note 4x	
	Pada gambar dapat terlihat bahwa model dapat mendeteksi 6 objek gambar susu bubuk dengan waktu inferensi yang dihasilkan 773ms.

Hasil Pengujian Pada Perangkat Xiaomi Redmi Note 4x	
	Pada gambar dapat terlihat bahwa model dapat mendeteksi 9 objek gambar susu bubuk dengan waktu inferensi yang dihasilkan 915ms.
Hasil Pengujian Pada Perangkat Oppo A53	
	Pada gambar dapat terlihat bahwa model dapat mendeteksi 8 objek gambar susu bubuk dengan waktu inferensi yang dihasilkan 602ms.

Hasil Pengujian Pada Perangkat Xiaomi Redmi Note 4x	
	Pada gambar dapat terlihat bahwa model dapat mendeteksi 9 objek gambar susu bubuk dengan waktu inferensi yang dihasilkan 631ms.

Pada Tabel 4.4 diketahui bahwa model dapat berjalan pada kedua perangkat *android* walaupun kemampuan mengenali objek dengan *bounding box* tidak sempurna, tapi cukup akurat. Jika diperhatikan dengan seksama, kemampuan mengenali objek juga bergantung pada kualitas kamera yang digunakan. Hal tersebut dapat terlihat jika membandingkan objek yang sama, perangkat Oppo lebih banyak menghasilkan *bounding box* serta waktu yang lebih cepat daripada perangkat Xiaomi. Selain itu, dari gambar diketahui bahwa sesekali model terlihat membutuhkan waktu hingga 900ms untuk melakukan deteksi. Waktu inferensi yang dihasilkan dari kedua perangkat ditunjukkan pada Tabel 4.5.

Tabel 4.5 Hasil *Evaluation* Setiap Label

Nama Perangkat	Waktu Inferensi
Xiaomi Redmi Note 4x	< 950ms
Oppo A53	< 650ms

Berdasarkan Tabel 4.5 disimpulkan bahwa model YOLOv4-*tiny* mampu melakukan *object detection* secara *real-time* pada produk susu bubuk, dengan rata-rata *inference time* yang diperlukan 600 hingga 700ms dari kedua perangkat. *Model mobile* ini tidaklah menjadi sistem

OSA yang utuh. Akan tetapi, menjadi alat yang dapat membantu proses deteksi sehingga pencatatan yang dapat dilakukan lebih cepat dan otomatis.

BAB V
KESIMPULAN DAN SARAN

Pada bab ini akan dibahas mengenai kesimpulan yang berhasil diperoleh selama melakukan penelitian. Selain itu, terdapat juga saran-saran penelitian yang dapat dijadikan acuan untuk pengembangan lebih lanjut pada penelitian selanjutnya.

5.1 Kesimpulan

Berdasarkan hasil dari penelitian yang telah dilakukan, peneliti dapat menarik beberapa kesimpulan. Beberapa poin kesimpulan sebagai berikut:

a. Penelitian ini berhasil mengimplementasikan model arsitektur YOLOv4-*tiny* untuk melakukan deteksi objek pada produk retail. Produk retail yang digunakan dalam kasus ini terdiri dari 106 *class* produk susu bubuk baik dus maupun kaleng.
b. Model arsitektur YOLOv4-*tiny* dipilih karena telah terbukti mampu melakukan deteksi objek dengan hasil yang cukup baik.
c. Model YOLOv4-*tiny* yang digunakan merupakan hasil dari *training* selama 46 jam dari 111 jam total waktu *training*.
d. Hasil performa yang didapatkan dari model pada penelitian ini cukup tinggi dengan nilai mAP menyentuh angka 92,12%, nilai *Precision* sebesar 0,78, nilai *Recall* sebesar 0,82, dan F1-*score* sebesar 0,80.
e. Berdasarkan pengujian yang dilakukan pada perangkat android didapatkan waktu inferensi rata-rata berkisar 600 hingga 700ms walaupun sesekali model membutuhkan waktu hingga 900ms. Waktu tersebut merupakan waktu yang cukup untuk menjalankan *object detection* secara *real-time* karena dapat berjalan kurang dari 1 detik.

5.2 Saran

Dalam penelitian ini masih terdapat kekurangan, yakni lamanya waktu yang diperlukan model pada saat melakukan *training dataset*. Oleh karena itu, peneliti berharap penelitian ini dapat dikembangkan lebih lanjut. Terdapat beberapa saran yang dapat diimplementasikan untuk pengembangan berikutnya. Beberapa saran tersebut sebagai berikut:

a. Menggunakan *dataset* yang memiliki resolusi lebih tinggi untuk melihat pengaruh resolusi terhadap performa yang dihasilkan dari model.
b. Menerapkan model arsitektur yang lebih ringan.

DAFTAR PUSTAKA

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Rajat, M., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). *TensorFlow: A System for Large-Scale Machine Learning*. USENIX Association.

Akbar, R., & Juliastrioza, J. (2015). Penerapan Enterprise Resource Planning (ERP) untuk Sistem Informasi Pembelian, Persedian dan Penjualan Barang pada Toko EMI GROSIR dan ECERAN. *Jurnal Nasional Teknologi Dan Sistem Informasi*, *1*(1), 7–17. https://doi.org/10.25077/TEKNOSI.v1i1.2015.7

Alfiah, E. N. (2021). *Pengendalian Internal Atas Persediaan Barang Dagang di Swalayan Sendy's Kota Palangka Raya*.

Amanda Putri, R. (2020). *Sistem Pakar*. http://repository.uinsu.ac.id/8610/

Amidi, A., & Amidi, S. (2019). *Convolutional Neural Networks cheatsheet*. Retrieved May 8, 2023. https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

Amrizal, V., & Aini, Q. (2013). *Kecerdasan Buatan*. Halaman Moeka Publishing.

Angelia, D. (2022). *Supermarket dengan Gerai Terbanyak di Indonesia 2021*. Retrieved April 27, 2023. https://goodstats.id/article/supermarket-dengan-gerai-terbanyak-di-indonesia-2021-QVGHG

Anjarwan, S. (2018). Pengaruh Kelengkapan Produk, Persepsi Harga dan Lokasi Terhadap Kepuasan Konsumen pada Toserba DM Baru 1 Pleret Bantul Yogyakarta. In *Jurnal Ekobis Dewantara* (Vol. 1, Issue 3).

Arandhea, A. S., & Puspitasari, R. (2021). Penerapan Sistem Informasi Akuntansi Untuk Persediaan Barang Dagang. *Jurnal Aplikasi Bisnis Kesatuan*, *1*(2). https://doi.org/10.37641/jabkes.v1i2.1180

Ardiansyah, M. N., Muttaqin, P. S., Prasetio, M. D., & Novitasari, N. (2021). Identifikasi Objek/Produk untuk Proses Stock Taking Barang menggunakan Konsep Object Recognition. *Jurnal Rekayasa Sistem & Industri (JRSI)*, *8*(01), 28. https://doi.org/10.25124/jrsi.v8i1.455

Badharudheen, J. K. (2021). *Vision System for Apple Harvesting Robot*. https://digikogu.taltech.ee/et/Download/4d27e33c-0c66-493e-a56a-6b02e1b9e4af

Bochkovskiy, A. (2021). Retrieved May 11, 2023. *Darknet*. https://github.com/AlexeyAB/darknet.git

Bodla, N., Singh, B., Chellappa, R., & Davis, L. S. (2017). Soft-NMS-Improving Object Detection With One Line of Code. *IEEE International Conference on Computer Vision (ICCV)*. https://doi.org/https://doi.org/10.48550/arXiv.1704.04503

Campo, K., Gijsbrechts, E., & Nisol, P. (2000). Towards understanding consumer response to stock-outs. *Journal of Retailing*, *76*(2), 219–242. https://doi.org/10.1016/S0022-4359(00)00026-9

Chaniago, H. (2021). *Manajemen Ritel dan Implementasinya*. Edukasi Riset Digital PT.

Cholissodin, I., Sutrisno, Soebroto, A. A., Hasanah, U., & Febiola, Y. I. (2019). *Buku Ajar AI, Machine Learning & Deep Learning*. https://www.researchgate.net/publication/348003841

Corsten, D., & Gruen, T. (2003). Desperately seeking shelf availability: an examination of the extent, the causes, and the efforts to address retail out-of-stocks. *International Journal of Retail & Distribution Management*, *31*(12), 605–617. https://doi.org/10.1108/09590550310507731

Dai, X., Spasic, I., Chapman, S., & Meyer, B. (2020). The State of the Art in Implementing Machine Learning for Mobile Apps: A Survey. *Conference Proceedings - IEEE SOUTHEASTCON*, *2020-March*. https://doi.org/10.1109/SoutheastCon44009.2020.9249652

Dewi, S. R. (2018). *Deep Learning Object Detection pada Video Menggunakan Tensorflow dan Convolutional Neural Network*.

Disemandi, H. S., & Ariesta Nadia, P. (2021). Produk Bahan Pangan Kadaluarsa yang Diperjualbelikan di Supermarket: Suatu Kajian Hukum Perlindungan Konsumen. *Maleo Law Journal*, *5*(2).

Ezhilkumar, M. (2020). Enhancing Behavioral Intention in Out-Of-Stock Situations-The Mediating Role of Perceived Product Uniqueness and Perceived Consumption Risk. In *Academy of Marketing Studies Journal* (Vol. 24, Issue 2).

Fauza, M. (2017). *Analisis Faktor yang Mempengaruhi Eksistensi Ritel Tradisional dalam Menghadapi Ritel Modern di Kecamatan Medan Amplas*.

Fina, C. A., Budiyati, H., & Rudatyo, H. (2020). Pengenalan Pola Isyarat Tangan pada Input Hand Gesture Dinamis. *IJCCS*, *x, No.x*, 1–5.

Firmansyah, R. (2021). *Implementasi Deep Learning Menggunakan Convolutional Neural Network Untuk Klasifikasi Bunga*.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Gujar, H., Mhatre, P., Ghanate, S., Kadam, S., Kurle, D., Shitole, S., & Chile, S. (2016). Python Based Image Processing. *Avishkar*. https://www.researchgate.net/publication/309208697_Python_Based_Image_Processing

Gunarso, B. (2013). *Artificial intelligence development center di Malang Tema smart building*. http://etheses.uin-malang.ac.id/1285/

Hafiz Ar, N., Kurniawan, M., & Surabaya, A. T. (2021). Analisis Fast Moving Consumer Goods untuk Memprakirakan Penjualan Barang Menggunakan Metode Triple Exponential Smoothing 115. *INTEGER: Journal of Information Technology*. https://doi.org/https://doi.org/10.31284/j.integer.2021.v6i2.2311

Haroon Shakeel. (2020). *Convert Darknet YOLOv4 or YOLOv3 to TensorFlow Model*. Retrieved May 17, 2023 https://github.com/haroonshakeel/tensorflow-yolov4-tflite

Harun, A., Mustakim, & Brilian, O. K. (2023). Implementasi Deep Learning Menggunakan Metode You Only Look Once untuk Mendeteksi Rokok. *Jural Media Informatika Budidarma*. https://doi.org/10.30865/mib.v7i1.5409

Hasibuan, Z. A. (2007). *Metodologi Penelitian pada Bidang Ilmu Komputer dan Teknologi Informasi*.

He, Y., Zeng, H., Fan, Y., Ji, S., & Wu, J. (2019). Application of Deep Learning in Integrated Pest Management: A Real-Time System for Detection and Diagnosis of Oilseed Rape Pests. *Mobile Information Systems*, *2019*. https://doi.org/10.1155/2019/4570808

Higa, K., & Iwamoto, K. (2019). Robust Shelf Monitoring Using Supervised Learning for Improving On-Shelf Availability in Retail Stores. *Sensors*, *19*(12), 2722. https://doi.org/10.3390/s19122722

Hijazi, S., Kumar, R., & Rowen, C. (2015). Using Convolutional Neural Networks for Image Recognition. *Cadence Design Systems Inc.* www.cadence.com

Hijriani, A., Safitri, J. A., Adi Pribadi, R. I., & Andrian, R. (2020). Pengembangan Sistem Informasi Manajemen Supplier dan Barang dengan Extreme Programming. *Jurnal Teknik Informatika Dan Sistem Informasi*, *6*(1). https://doi.org/10.28932/jutisi.v6i1.2132

Imantiyar, R. (2021). *Pengaruh Bias Dataset Terhadap Performa Akurasi Deteksi Objek pada Arsitektur Efficientdet-Lite*.

Iswandi, M., & Ester, S. (2020). Pengaruh Kepercayaan Merek, Kesadaran Merek dan Persepsi Kualitas Terhadap Keputusan Pembelian Pelanggan di Supermarket Gelael Mt Haryono. *JURNAL GICI*, *12*.

Jalled, F., & Voronkov, I. (2016). Object Detection using Image Processing. *ArXiv*. http://arxiv.org/abs/1611.07791

Jannah, Z. S., & Sutanto, F. A. (2022). Implementasi Algoritma YOLO (You Only Look Once) Untuk Deteksi Rias Adat Nusantara. *Jurnal Ilmiah Universitas Batanghari Jambi*, *22*(3), 1490. https://doi.org/10.33087/jiubj.v22i3.2421

Jdid, B., Hassan, K., Dayoub, I., Lim, W. H., & Mokayef, M. (2021). Machine Learning Based Automatic Modulation Recognition for Wireless Communications: A Comprehensive Survey. *IEEE Access*, *9*, 57851–57873. https://doi.org/10.1109/ACCESS.2021.3071801

Jha, D., Mahjoubfar, A., & Joshi, A. (2022). *Designing an Efficient End-to-end Machine Learning Pipeline for Real-time Empty-shelf Detection*. http://arxiv.org/abs/2205.13060

Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A Review of Yolo Algorithm Developments. *Procedia Computer Science*, *199*, 1066–1073. https://doi.org/10.1016/j.procs.2022.01.135

Jiang, Z., Zhao, L., Li, S., Jia, Y., & Liquan, Z. (2020). Real-time object detection method for embedded devices. *ArXiv*.

KBBI. (2016a). Produk. *Kamus Besar Bahasa Indonesia*. Retrieved March 8, 2023. https://kbbi.kemdikbud.go.id/entri/produk

KBBI. (2016b). Retail. *Kamus Besar Bahasa Indonesia*. Retrieved March 8, 2023. https://kbbi.kemdikbud.go.id/entri/retail

Khairunnas, Yuniarno, E. M., & Zaini, A. (2021). Pembuatan Modul Deteksi Objek Manusia Menggunakan Metode YOLO untuk Mobile Robot. *Jurnal Teknik ITS*, *10*.

Khatami, M. S. (2022). *Deteksi Kendaraan Menggunakan Algoritma You Only Look Once (Yolo) V3*.

Limanto, S. (2018, March 8). Pengembangan Aplikasi Sistem Informasi untuk Membantu Mengontrol Stok dan Meningkatkan Layanan kepada Pelanggan. *Konferensi Nasional Sistem Informasi*.

Limanto, S., Tjandra, E., & Indrawan, A. (2019). Rekomendasi Pembelian Barang Pada Sistem Retail Dengan Metode Dekomposisi Census II. *Teknika*, *8*(2), 126–132. https://doi.org/10.34148/teknika.v8i2.222

Lina, Q. (2019). *Apa itu Convolutional Neural Network?*. Retrieved May 2, 2023. https://medium.com/@16611110/apa-itu-convolutional-neural-network-836f70b193a4

Liu, Q., Fan, X., Xi, Z., Yin, Z., & Yang, Z. (2022). Object detection based on Yolov4-Tiny and Improved Bidirectional feature pyramid network. *Journal of Physics: Conference Series*, *2209*(1). https://doi.org/10.1088/1742-6596/2209/1/012023

Marifatul Azizah, L., Fadillah Umayah, S., & Fajar, F. (2018). Deteksi Kecacatan Permukaan Buah Manggis Menggunakan Metode Deep Learning dengan Konvolusi Multilayer. *Semesta Teknika*, *21*(2). https://doi.org/10.18196/st.212229

Markoulidakis, I., Rallis, I., Georgoulas, I., Kopsiaftis, G., Doulamis, A., & Doulamis, N. (2021). Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem. *Technologies*, *9*(4). https://doi.org/10.3390/technologies9040081

Mccarthy, J. (2007). *What is Artificial Intelligence?*. Retrieved May 2, 2023. http://www-formal.stanford.edu/jmc/

Melek, C. G., Sonmez, E. B., & Albayrak, S. (2019). Object Detection in Shelf Images with YOLO. *IEEE EUROCON 2019 -18th International Conference on Smart Technologies*. https://doi.org/https://doi.org/10.1109/EUROCON.2019.8861817

Milella, A., Petitti, A., Marani, R., Cicirelli, G., & D'Orazio, T. (2020). Towards Intelligent Retail: Automated on-Shelf Availability Estimation Using a Depth Camera. *IEEE Access*, *8*, 19353–19363. https://doi.org/10.1109/ACCESS.2020.2968175

Mitchell, A. (2012). Improving On-Shelf Availability It Matters More. *SymphonyIRI Group*. www.SymphonyIRI.eu

Mohamed, Z. E. (2019). Using the artificial neural networks for prediction and validating solar radiation. *Journal of the Egyptian Mathematical Society*, *27*(1), 47. https://doi.org/10.1186/s42787-019-0043-8

Mubarok, H. (2019). *Identifikasi Ekspresi Wajah Berbasis Citra Menggunakan Algoritma Convolutional Neural Network (CNN)*.

Muningsih, E., & Kiswati, D. S. (2015). Penerapan Metode K-Means Untuk Clustering Produk Online Shop Dalam Penentuan Stok Barang. *Jurnal Bianglala Informatika*, *3*. lppm3.bsi.ac.id/jurnal

Naufal, M. F., & Kusuma, S. F. (2021). Pendeteksi Citra Masker Wajah Menggunakan CNN dan Transfer Learning. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *8*(6), 1293–1300. https://doi.org/10.25126/jtiik.202185201

Normawati, D., & Prayogi, S. A. (2021). Implementasi Naïve Bayes Classifier Dan Confusion Matrix Pada Analisis Sentimen Berbasis Teks Pada Twitter. *Jurnal Sains Komputer & Informatika (J-SAKTI)*, *5*(2), 697–711.

Nurkamid, M., & Sutejo. (2010). Metode Kecerahan Citra Kontras Citra dan Penajaman Citra untuk Peningkatan Mutu Citra. *ArXiv*. https://api.semanticscholar.org/CorpusID:1854780

O'shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. *ArXiv*.

Padilla, R., Netto, S. L., & da Silva, E. A. B. (2020). A Survey on Performance Metrics for Object-Detection Algorithms. *International Conference on Systems, Signals and Image Processing (IWSSIP)*, 237–242. https://doi.org/10.1109/IWSSIP48289.2020.9145130

Padilla, R., Netto, S. L., & Silva, E. A. B. da. (2020). A Survey on Performance Metrics for Object-Detection Algorithms. *International Conference on Systems, Signals and Image Processing (IWSSIP)*. https://doi.org/10.1109/IWSSIP48289.2020.9145130

Paolanti, M., Romeo, L., Martini, M., Mancini, A., Frontoni, E., & Zingaretti, P. (2019). Robotic retail surveying by deep learning visual and textual data. *Robotics and Autonomous Systems*, *118*, 179–188. https://doi.org/10.1016/j.robot.2019.01.021

Puspa, R., Permana, A., Karunia, E., Ekonomi, F., Bisnis, D., Bina, U., & Serang Banten, B. (2020). Faktor yang Mempengaruhi Kepuasan Pelanggan Berdasarkan Bauran Pemasaran pada Supermarket K-Store Krakatau Junction 1. *Jurnal Bina Bangsa Ekonomika (JBBE)*, *13*(02). https://doi.org/10.46306/jbbe.v13i2

Rahmasari, T. (2019). Perancangan Sistem Informasi Akuntansi Persediaan Barang Dagang Pada Toserba Selamat Menggunakan Php Dan Mysql. *Is The Best Accounting Information Systems and Information Technology Business Enterprise This Is Link for OJS Us*, *4*(1), 411–425. https://doi.org/10.34010/aisthebest.v4i1.1830

Rahmatullah, I. L. (2022). *Pengenalan Suara Menggunakan Algoritma Convolutional Neural Network pada Gim Pembelajaran Bahasa Arab*.

Ranjan, J., & Puri, S. (2012). Out of Stock conditions affecting Customer satisfaction and customer loyalty. In *Journal of Business and Retail Management Research (JBRMR)* (Vol. 6). ABRM.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. http://arxiv.org/abs/1506.02640

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., & Savarese, S. (2019). Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Rosebrock, A. (2016). *Intersection over Union (IoU) for object detection*. Retrieved March 8, 2023. https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/

Rumiarti, C. D., Setiawan, B. R., Dewa, I., & Wiana, M. P. (2019). Kajian Perencanaan Strategis Sistem Informasi pada Bisnis Ritel Berbasis Metodologi WARD & PEPPARD: Studi Kasus PT. Gramedia Asri Media. *Jurnal Teknologi Informasi Dan Ilmu Komputer (JTIIK)*, *6*(3), 245–256. https://doi.org/10.25126/jtiik.20196926

Saqlain, M., Rubab, S., Khan, M. M., Ali, N., & Ali, S. (2022). Hybrid Approach for Shelf Monitoring and Planogram Compliance (Hyb-SMPC) in Retails Using Deep Learning and Computer Vision. *Mathematical Problems in Engineering*, *2022*. https://doi.org/10.1155/2022/4916818

Saragih, C. V. B., Shihab, M. S., & Purwanto, W. (2013). Pengaruh Kualitas Produk, Ketersediaan Produk dan Gaya Hidup Terhadap Keputusan Pembelian Produk Lulur Mandi Sumber Ayu di Jakarta. *Jurnal MIX*, *6*(2). 01/10/2022http://digilib.mercubuana.ac.id/manager/t!@file_artikel_abstrak/Isi_Artikel_949723712677.pdf

Saresa, L., Afriani, S., & Fitriano, Y. (2021). Analisis Sistem Pengendalian Internal Persediaan Barang Dagang pada Alfamart Merapi Kebun Tebeng Bengkulu. *Jurnal Ekonomi, Manajemen, Akuntansi Dan Keuangan*, *2*(3).

Šećerović, L., & Papić, V. (2018). Detecting missing products in commercial refrigerators using convolutional neural networks. *Symposium on Neural Networks and Application (NEUREL)*.

Selvam, P., Abraham, J., & Koilraj, S. (2022). A Deep Learning Framework for Grocery Product Detection and Recognition. *Food Analytical Methods*. https://doi.org/10.21203/rs.3.rs-1431986/v1

Sinha, A., Banerjee, S., & Chattopadhyay, P. (2022). An Improved Deep Learning Approach For Product Recognition on Racks in Retail Stores. *ArXiv*. http://arxiv.org/abs/2202.13081

Son, J., Kang, J. H., & Jang, S. (2019). The effects of out-of-stock, return, and cancellation amounts on the order amounts of an online retailer. *Journal of Retailing and Consumer Services*, *51*, 421–427. https://doi.org/10.1016/j.jretconser.2019.02.008

Suryadarma, D., Poesoro, A., Budiyati, S., Akhmadi, & Rosfadhila, M. (2007). *Dampak Supermarket terhadap Pasar dan Pedagang Ritel Tradisional di Daerah Perkotaan di Indonesia*. www.smeru.or.id

Swasono, M. A., & Prastowo, A. T. (2021). Analisis dan Perancangan Sistem Infomasi Pengendalian Persediaan Barang. *Jurnal Informatika Dan Rekayasa Perangkat Lunak (JATIKA)*.

Tasnim, A., Saiduzzaman, Md., Rahman, M. A., Akhter, J., & Rahaman, A. S. Md. M. (2022). Performance Evaluation of Multiple Classifiers for Predicting Fake News. *Journal of Computer and Communications*, *10*(09), 1–21. https://doi.org/10.4236/jcc.2022.109001

Tjahyanti, L. P. A. S., Saputra, P. S., & Gitakarma, M. S. (2022). Peran Artificial Intelligence (AI) untuk Mendukung Pembelajaran di Masa Pandemi Covid-19. *Jurnal Komputer DanTeknologi Sains (KOMTEKS)*.

Wells, R. (2014). *Pengolahan Citra Digital*.

Wijaya, T. A., & Prayudi, Y. (2010). Implementasi Visi Komputer dan Segmentasi Citra untuk Klasifikasi Bobot Telur Ayam Ras. *Seminar Nasional Aplikasi Teknologi Informasi*.

Winston, P. H., & Prendergast, K. A. (1986). *The AI business commercial uses of artificial intelligence*. MIT Press.

Wulandari, E., Mulyatini, N., & Yustini, I. (2020). Pengaruh on Shelf Availability dan Gaya Hidup Terhadap Keputusan Pembelian (Suatu Studi pada Konsumen Toko H. Junaedi Kawali Distributor Produk Merek Rabbani). *Bussiness Management And Entrepreneurship Jurnal*, *2*.

Yani, M., Irawan, B., & Setiningsih, C. (2019). Application of Transfer Learning Using Convolutional Neural Network Method for Early Detection of Terry's Nail. *Journal of Physics: Conference Series*, *1201*(1), 012052. https://doi.org/10.1088/1742-6596/1201/1/012052

Yilmazer, R., & Birant, D. (2021). Shelf auditing based on image classification using semi-supervised deep learning to increase on-shelf availability in grocery stores. *Sensors (Switzerland)*, *21*(2), 1–26. https://doi.org/10.3390/s21020327

Yuswandi, D., & Supriyanto, H. (2021). Analisis Perbaikan Kualitas Layanan Penjualan Menggunakan Metode SERVQUAL dan TRIZ untuk Menciptakan Loyalitas Konsumen pada CV. XYZ. *Prosiding Seminar Nasional Sains Dan Teknologi Terapan*, *9*.

Zhang, J. (2019). Basic Neural Units of the Brain: Neurons, Synapses and Action Potential. *ArXiv*. http://arxiv.org/abs/1906.01703

Zhiqiang, W., & Jun, L. (2017). A Review of Object Detection Based on Convolutional Neural Network. *Proceedings of the 36th Chinese Control Conference*.

Zulkiflie, M. A. (2021). *Implementasi Algoritma Object Detection Yolov4 dan Euclidean Distance dalam Mendeteksi Pelanggaran Social Distancing*.

LAMPIRAN