

RANGKUMAN HASIL REVISI

Nama : Gabriel Advent Batan

NIM : 205314096

Judul TA : Pengenalan Objek Untuk Pembelajaran Anak-Anak

Menggunakan Arsitektur YOLO

No.	Revisi	Tanggapan	Nomor Halaman
1.	Judul yang dipresentasikan menghilangkan aspek ‘informatika’	Pada halaman judul telah di ubah kata algoritma menjadi arsitektur	Revisi_1
2.	Apa perbedaan YOLO dan CNN	Penjelasan mengenai perbedaan YOLO dan CNN sudah dimasukkan ke dalam bab II di sub 2.2.5 pada paragraf ke 2 dan 3	Revisi_2
3.	“YOLO dapat digunakan untuk mendeteksi objek dalam waktu yang lebih singkat dibandingkan dengan CNN khususnya YOLO agar dapat membantu anak-anak untuk mengenali objek sekitar”	YOLO dapat digunakan untuk mendeteksi objek dengan kecepatan lebih tinggi secara <i>real-time</i> dibandingkan metode deteksi objek berbasis CNN. Hal ini telah dibuktikan dengan beberapa penelitian sebelumnya yang telah dilakukan. Hal ini dapat dilihat pada Revisi_3 terkait <i>review literature</i> . Selain itu, berdasarkan penelitian-penelitian tersebut, dapat dipastikan bahwa YOLO dapat digunakan untuk mendeteksi objek baik secara <i>real-time</i> maupun <i>non-real-time</i> dan berkaitan dengan penelitian ini dapat dilihat bahwa YOLO dapat digunakan untuk membantu anak-anak dalam mengenali objek sekitar. Hal ini dapat dilihat pada Revisi_3b mengenai kesimpulan dari penelitian ini.	Revisi_3 dan Revisi_3b

4.	Bagaimana penerapan computer vision (jelaskan teknologi computer vision)	Penjelasan mengenai computer vision sudah dimasukkan ke dalam bab II pada sub 2.2.4 mengenai computer vision.	Revisi_4
5a.	Penjelasan langkah demi langkah arsitektur YOLO	Penjelasan mengenai langkah-langkah arsitektur telah dimasukan ke dalam poin 3.7 di bab III	Revisi_5a
5b.	Jelaskan kegunaan tiap parameter	Sudah dimasukkan ke dalam tabel skenario pengujian pada poin 3.8 di bab III	Revisi_5b
6.	Jelaskan hasil dari roboflow	Hasil dari roboflow telah dijelaskan pada poin 4.3 mengenai import dataset di bab IV. Hasil dari ekstrasi dataset tersebut merupakan hasil yang telah dilakukan di roboflow.	Revisi_6
7.	Alasan pemilihan objek perlu diberikan, mengingat sasaran usia 2-4 tahun	Alasan pemilihan objek tersebut telah dimasukkan ke dalam poin 3.2 mengenai data di bab III	Revisi_7
8.	Dependency dalam modelling apa saja yang dilakukan	Untuk proses mendapatkan dependency secara rinci telah ditambahkan ke dalam poin 4.4.1 di bab IV	Revisi_8
9.	Pengguna user interface, siapa respondennya	Terkait pengguna user interface telah dimasukkan ke dalam poin 4.6.1 di bab IV dan untuk responen pun telah dimasukkan ke dalam poin 3.9.1 mengenai target kuesioner di bab III	Revisi_9a dan Revisi_9b
10.	Gambar 4.20 belum dijelaskan untuk setiap grafik	Penjelasan terkait grafik sudah dimasukkan ke dalam tabel 4.4 di bab IV	Revisi_10
11.	Bagaimana cara kerja aplikasi/website	Penjelasan terkait cara kerja aplikasi telah dimasukkan ke dalam poin 4.6.4 di bab IV	Revisi_11

12.	Apa benefit dari aplikasi/website	Manfaat dari website/aplikasi sudah dimasukkan ke dalam poin 4.6.5 pada bab IV	Revisi_12
13.	Tujuan dan masalahnya apa? Membantu anak-anak atau orangtua?	Terkait hal ini, untuk rumusan masalah telah diubah menjadi “.... Bagaimana penerapan teknologi <i>computer vision</i> khususnya menggunakan YOLO agar dapat membantu orang tua dalam mengajarkan kepada anak-anak dalam pengenalan objek sekitar?”	Revisi_13
14.	Di bab 2 tuliskan semua apa yang dikerjakan	Hyperparameter yang digunakan di skenario telah dimasukkan ke dalam poin 2.2.6	Revisi_14
15.	Gunakan suara untuk menunjukan hasil (di website)	Penggunaan suara untuk menunjukkan hasil telah diimplementasi ke dalam code yang dapat dilihat pada implementasi code untuk deteksi di poin 4.6.2	Revisi_15
16.	Gunakan gambar yang sesuai dengan kelas yang digunakan	Gambar yang digunakan baik untuk proses pelatihan, validasi, dan testing merupakan gambar yang telah disesuaikan dengan kelas-kelasnya. Hal ini dapat dilihat pada poin 3.2 terkait dengan data pada bab III.	Revisi_16
17.	Model pre-trained dengan menggunakan custom dataset (hal. 44)	Penjelasan ini telah dihilangkan dan dimasukkan ke dalam bagian skenario pengujian pada poin 3.8 di bab III	Revisi_17

Rangkuman Revisi Kedua

No.	Revisi	Tanggapan	Nomor Halaman
1.	Perbaiki pustaka (menggantikan jurnal Journal of Metaverse Computer Vision)	Untuk judul jurnal sudah diganti dari “Journal of Metaverse Computer Vision” menjadi “Computer Vision in the Metaverse”	revisi baru 1
2.	Perbaikan format penulisan: 1. Dokumen dibuat rata kiri dan sub bab tidak dibuat menjorok 2. Penomoran halaman pada skripsi	1. Terkait tulisan rata kiri sudah diubah dan sub bab tidak di buat menjorok 2. Penomoran halaman telah disesuaikan dengan pedoman skripsi, di mana nomor halaman bab berada di tengah bawah dan berikutnya ada di tengah atas	1. revisi baru 2 a 2. revisi baru 2 b
3.	Pra-trained atau pre-trained	Kesalahan tulisan pre-trained sudah diubah dari “pra-trained” menjadi “pre-trained”	revisi baru 3
4.	Penggunaan istilah yang konsisten: 1. Bounding box atau kotak pembatas 2. Loss atau kerugian 3. Precision atau presisi 4. Code atau kode	Penggunaan kata yang konsisten sudah diperbaiki menjadi: 1. Bounding box 2. Loss 3. Precision 4. Code	Contoh perubahan: 1. revisi baru 3 a 2. revisi baru 3 b 3. revisi baru 3 c 4. revisi baru 3 d
5.	Mengubah kata “projek” menjadi “proyek”	Kata projek telah diubah menjadi proyek	Salah satu contoh: revisi baru 4
6.	kelas yang dilatih ada berapa ? bab 3.2 dan batasan masalah berbeda. kenapa hanya 6 kelas ?	Terjadi kesalahan penulisan pada sub 3.2 mengenai data. Untuk kelas yang dilatih ada 6 kelas, yaitu <i>handphone</i> , mobil, tas, manusia, jam, dan sepatu, seperti yang ada pada batasan masalah. Karena itu, pada poin 3.2 telah disesuaikan kelasnya seperti pada batasan masalah.	revisi baru 5
7.	Tampilan jika menggunakan handphone dan hasil deteksi selain orang	Untuk tampilan jika menggunakan handphone atau ponsel sudah dimasukkan. Tampilan yang diambil adalah tampilan awal ketika website diakses dan tampilan ketika menggunakan salah satu menu.	revisi baru 7
8.	Perbaikan abstrak	Abstrak telah diperbaiki agar berisi masalah yang diteliti, metodologi yang digunakan, hasil penelitian, kesimpulan dan saran.	revisi baru 8
9.	Kata “percobaan ketujuh”	Percobaan ketujuh merupakan eksperimen ketujuh yang dilakukan. Sebagai bentuk perbaikan, kata eksperimen diubah menjadi percobaan.	revisi baru 9

**PENGENALAN OBJEK UNTUK PEMBELAJARAN ANAK-ANAK
MENGGUNAKAN ARSITEKTUR YOLO**

SKRIPSI

Diajukan untuk memenuhi salah satu syarat

memperoleh Gelar Sarjana Komputer

Program Studi Informatika



Disusun oleh:

Gabriel Advent Batan

NIM: 205314096

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS SANATA DHARMA

YOGYAKARTA

2024

HALAMAN PERSETUJUAN PEMBIMBING

SKRIPSI

**PENGENALAN OBJEK UNTUK PEMBELAJARAN ANAK-ANAK
MENGGUNAKAN ARSITEKTUR YOLO**

Disusun oleh:

Gabriel Advent Batan

NIM: 205314096

Dosen Pembimbing,

(Drs. Hari Suparwito, S.J., M.App.IT)

(tanggal persetujuan)

ABSTRAK

Pentingnya periode usia dini sebagai waktunya anak terhadap rangsangan menjadi dasar bagi pendekatan inovatif dalam pembelajaran. Perkembangan teknologi telah memungkinkan pembelajaran interaktif, namun pemanfaatannya pada usia dini masih kurang optimal. Penelitian ini mengaitkan konsep pengenalan objek dalam *computer vision*, khususnya melalui algoritma *You Only Look Once* (YOLO), dengan konteks pembelajaran anak usia dini. YOLO, sebagai pendekatan integratif deteksi dan klasifikasi objek, telah menjadi populer dalam pengenalan objek *real-time*. Penelitian ini mencoba mengoptimalkan teknologi *computer vision*, khususnya dengan implementasi YOLO, dalam pemahaman lingkungan sekitar untuk pembelajaran anak usia dini. Penelitian ini melibatkan beberapa parameter-parameter yang digunakan dalam YOLO. Dari 32 percobaan berdasarkan kombinasi parameter, diketahui bahwa percobaan ketujuh mampu menghasilkan mAP yang lebih tinggi dari percobaan-percobaan yang lain. Percobaan ketujuh ini mampu menghasilkan mAP sebesar 0,87916 atau 87%. Selanjutnya dengan bantuan Streamlit, akan dibuat *website* untuk diujicobakan dan disebarluaskan bersamaan dengan kuesioner yang sudah diuji dengan uji validitas dan reliabilitas. Dari 27 responden kuesioner ini akan dihitung berdasarkan aspek *usability* dan mendapatkan hasil sangat layak. Hal ini mampu membuktikan bahwa pengimplementasian YOLO ke dalam *website* dapat membantu tumbuh kembang anak.

Kata kunci: YOLO, *object detection*, *computer vision*, usia dini, pendidikan

ABSTRACT

The importance of early years as a sensitive period for children's stimulation forms the basis for innovative approaches in education. Technological advancements have enabled interactive learning, yet its utilization in early childhood remains suboptimal. This research connects the concept of object recognition in computer vision, particularly through the You Only Look Once (YOLO) algorithm, with the context of early childhood education. The implementation of YOLO in this study involved optimizing specific parameters. Among 32 experiments based on parameter combinations, the seventh experiment yielded the highest mAP, with a value of 0.87916 or 87%. This experiment demonstrated that the YOLO algorithm is effective in optimizing computer vision technology for early childhood learning. Subsequently, a website was created using Streamlit, which was tested and distributed along with a questionnaire that had been validated for reliability. From 27 questionnaire respondents, the usability analysis results indicated that the website is highly suitable for use. This study proves that implementing YOLO into a website can aid children's development by helping them understand their surroundings. It is recommended to further develop the integration of this technology with various other interactive learning methods to maximize the potential of early childhood education.

Keywords: **YOLO, object detection, computer vision, early childhood, education.**

DAFTAR ISI

HALAMAN JUDUL.....	Error! Bookmark not defined.
HALAMAN PERSETUJUAN PEMBIMBING	ii
ABSTRAK	iii
ABSTRACT	iv
DAFTAR ISI.....	v
DAFTAR TABEL.....	ix
DAFTAR GAMBAR	x
DAFTAR LAMPIRAN.....	xiii
BAB I: PENDAHULUAN.....	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	3
1.3. Batasan Masalah.....	3
1.4. Tujuan Penelitian	3
1.5. Manfaat Penelitian	4
1.6. Sistematika Penulisan	4
BAB II: TINJAUAN PUSTAKA.....	6
2.1. Review Literature.....	6
2.2. Landasan Teori.....	13
2.2.1. Pendidikan Anak Usia Dini	13
2.2.2. Image Processing	14

2.2.3. Augmentasi	15
2.2.4. Computer Vision.....	16
2.2.5. You Only Look Once (YOLO).....	16
2.2.6. Hyperparameter YOLOv8	19
2.2.7. Roboflow	21
2.2.8. Labeling Gambar	21
2.2.9. Confusion Matrix.....	22
2.2.10. Mean Average Precision.....	22
2.2.11. Skala Likert.....	22
2.2.12. Uji Validitas.....	23
2.2.13. Uji Reliabilitas.....	24
2.2.14. Uji Kegunaan (Usability).....	24
BAB III METODE PENELITIAN.....	26
3.1. Gambaran Umum Penelitian	26
3.2. Data	27
3.3. Pre-Processing.....	28
3.3.1. Labeling Gambar	28
3.3.2. Resize Image.....	29
3.3.3. Augmentasi	29
3.3.4. Data Split	30
3.4. Mengimport Dataset Ke Proyek.....	30

3.5.	Modeling	30
3.5.1.	Mendapatkan Dependency Yolo Versi 8.....	30
3.5.2.	Pelatihan Model.....	31
3.6.	Evaluate.....	31
3.7.	Langkah – Langkah atau Alur Arsitektur YOLO	31
3.7.1.	Backbone dan Neck	31
3.7.2.	Head.....	37
3.8.	Skenario Pengujian.....	39
3.9.	Pengujian Kelayakan Kuesioner dan Kegunaan Hasil Akhir	40
3.9.1.	Target Kuesioner	40
3.8.2.	Waktu dan Tempat Pengujian	41
3.8.3.	Uji Kelayakan Kuesioner	41
	BAB IV HASIL PENELITIAN DAN PEMBAHASAN	44
4.1.	Pengumpulan Data	44
4.2.	Pre-processing Data	44
4.2.1.	Labeling Gambar.....	45
4.2.2.	Resize Image.....	46
4.2.3.	Augmentasi Image	46
4.2.4.	Data Split.....	49
4.3.	Impor Dataset.....	49
4.4.	Modeling	51

4.4.1. Mendapatkan Depedency YOLO	51
4.4.2. Pelatihan Model	52
4.5. Analisis Hasil Pengujian	56
4.6. Implementasi Aplikasi Deteksi Objek	65
4.6.1. Pengguna User Inteface.....	65
4.6.2. Interface Aplikasi	65
4.6.3. Implementasi Code untuk Deteksi	69
4.6.4. Alur atau Cara Kerja Aplikasi.....	75
4.6.5. Manfaat Dari Aplikasi.....	80
4.7. Pengujian Kegunaan Aplikasi	81
4.7.1. Uji Validitas	81
4.7.2. Uji Reliabilitas	83
4.7.3. Uji Usability	83
BAB V PENUTUP.....	87
5.1. Kesimpulan	87
5.2. Saran.....	87
DAFTAR PUSTAKA	89
LAMPIRAN	101

DAFTAR TABEL

Tabel 2. 1: Tabel review literature	8
Tabel 3. 1: Tabel Skenario Pengujian	40
Tabel 3. 2: Rincian pernyataan kuesioner	41
Tabel 3. 3: Ketentuan skala likert	42
Tabel 3. 4: Tingkat reliabilitas Cronbach's Alpha.....	42
Tabel 3. 5: Kategori Kegunaan	43
Tabel 4. 1: Daftar kombinasi parameter.....	53
Tabel 4. 2: Hasil pelatihan model	56
Tabel 4. 3: Kombinasi yang menghasilkan mAP terbaik.....	57
Tabel 4. 4: Tabel penjelasan dari grafik 4.28.....	61
Tabel 4. 5: Tabel narasi use case akses awal	75
Tabel 4. 6: Tabel narasi use case menu gambar.....	76
Tabel 4. 7: Tabel narasi use case menu video.....	77
Tabel 4. 8: Tabel narasi use case menu video.....	79
Tabel 4. 9: Tabel narasi use case real-time	79
Tabel 4. 10: Hasil kuesioner pengujian.....	81
Tabel 4. 11: Hasil perhitungan rHitung	82
Tabel 4. 12: Hasil uji validitas	82
Tabel 4. 13: Rincian responden.....	84
Tabel 4. 14: Tabel hasil kuesioner	84
Tabel 4. 15: Tabel hasil perhitungan usability	85

DAFTAR GAMBAR

Gambar 2. 1: Grafik perbandingan beberapa arsitektur berdasarkan mAP dan waktu	7
Gambar 2. 2: Ilustrasi proses digitalisasi	15
Gambar 2. 3: Ilustrasi proses augmentasi	15
Gambar 2. 4: Ilustrasi nilai yang merepresentasikan bounding box	18
Gambar 3. 1: Flowchart penelitian.....	26
Gambar 3. 2: Flowchart pre-processing	26
Gambar 3. 3: Flowchart Modelling.....	26
Gambar 3. 4: Gambar kumpulan dataset sesuai kelas.....	27
Gambar 3. 5: Contoh gambar yang telah dilakukan labeling.....	28
Gambar 3. 6: Ilustrasi nilai bounding box.....	28
Gambar 3. 7: Contoh hasil rescale	29
Gambar 3. 8: Contoh augmentasi yang diterapkan pada citra	29
Gambar 3. 9: Contoh confussion matrix	31
Gambar 3. 11: Output channel 2,2	32
Gambar 3. 12: Output channel 2,1	32
Gambar 3. 13: Output channel 1,3	32
Gambar 3. 14: Output channel 1,2	32
Gambar 3. 15: Output channel 1,1	32
Gambar 3. 10: Contoh matrix inputan 5x5 dan kernel 3x3	32
Gambar 3. 16: Hasil convolution	33
Gambar 3. 17: Output channel 3,3	33
Gambar 3. 18: Output channel 3,2	33

Gambar 3. 19: Output channel 3,1	33
Gambar 3. 20: Output channel 2,3	33
Gambar 3. 21: Hasil padding dengan nilai 0.....	34
Gambar 3. 22: Proses dari max pooling.....	34
Gambar 3. 23: Hasil dari max pooling 2x2	35
Gambar 3. 24: Hasil flatten.....	35
Gambar 3. 25: Hasil softmax	37
Gambar 3. 26: Proses pembagian gambar menjadi grid SxS	38
Gambar 3. 27: Ilustrasi proses deteksi YOLO	39
Gambar 4. 1: Proses upload data ke Roboflow	44
Gambar 4. 2: Proses annotate di Roboflow.....	45
Gambar 4. 3: Data hasil labeling atau bounding box	45
Gambar 4. 4: Proses resize pada dataset	46
Gambar 4. 5: Proses augmentasi dataset	47
Gambar 4. 6: Proses exposure dataset.....	47
Gambar 4. 7: Proses rotation dataset.....	48
Gambar 4. 8: Proses flip dataset.....	48
Gambar 4. 9: Proses pembagian dataset.....	49
Gambar 4. 10: Code untuk mengimpor dataset dari Roboflow	49
Gambar 4. 11: Keterangan jika berhasil mengunduh dan mengekstrak dataset ...	50
Gambar 4. 12: Hasil ekstrasi dataset	50
Gambar 4. 13: Isi dari file data.yaml.....	51
Gambar 4. 14: Code untuk menginstall library	51
Gambar 4. 15: Code untuk mengecek ultralytics	51

Gambar 4. 16: Output dari code untuk mengecek ultralytics.....	52
Gambar 4. 17: Mengimpor YOLO ke dalam projek	52
Gambar 4. 18: Code untuk mengombinasikan parameter	52
Gambar 4. 19: Code untuk melakukan pelatihan model pada YOLO	54
Gambar 4. 20: Contoh code yang disimpan di variabel array commands	54
Gambar 4. 21: Output dari perintah 'print(command)'	55
Gambar 4. 22: Code untuk menjalankan pelatihan sesuai dengan isi dari variabel commands	55
Gambar 4. 23: Output pelatihan tiap epoch	55
Gambar 4. 24: Code untuk melakukan validasi	57
Gambar 4. 25: Perbandingan optimizer	57
Gambar 4. 26: Grafik precision dan recall	58
Gambar 4. 27: Confusion matrix percobaan ketujuh	59
Gambar 4. 28: Kumpulan grafik hasil pelatihan model	60
Gambar 4. 29: Code untuk menjalankan testing	62
Gambar 4. 30: Gambar yang diprediksi dengan benar.....	63
Gambar 4. 31: Gambar yang kurang tepat diprediksi	64
Gambar 4. 32: Tampilan halaman pertama ketika website diakses	66
Gambar 4. 33: Tampilan halaman jika memilih menu gambar.....	66
Gambar 4. 34: Tampilan jika memilih menu video	67
Gambar 4. 35: Tampilan jika memilih menu youtube	67
Gambar 4. 36: Tampilan jika menggunakan atau ponsel.....	68
Gambar 4. 37: Tampilan ketika memilih menu Real-Time	68
Gambar 4. 38: Chart rincian responden	84

DAFTAR LAMPIRAN

Lampiran 1: Gambar confusion matrix	101
Lampiran 2: Capture code dari function showDetectFrame	102
Lampiran 3: Capture code dari function play_youtube.....	103
Lampiran 4: Capture code dari function live	103
Lampiran 5: Capture code dari class VideoTransformer	104
Lampiran 6: Capture code dari function process_uploaded_video.....	105
Lampiran 7: Capture code dari function play_stored_video.....	106
Lampiran 8: Capture code dari function take_picture	106
Lampiran 9: Capture code dari function up_picture	107
Lampiran 10: Daftar pertanyaan dalam kuesioner	108

BAB I:**PENDAHULUAN****1.1. Latar Belakang**

Pendidikan anak usia dini merupakan periode perkembangan yang penting dalam kehidupan anak (Heri Pratikno dkk., 2023; Musdalifah dkk., 2020; Supriadi dkk., 2021; Yuni Wulandari dkk., 2022). Pendidikan sejak usia dini turut mengambil peran dalam kesuksesan di masa depan anak. Hal ini juga ditekankan dalam undang-undang sistem pendidikan nasional Republik Indonesia no. 20 tahun 2003 pada bab 1 butir ke-14 (Heri Pratikno dkk., 2023). Usia dini merupakan waktu masa anak mulai peka dalam menerima rangsangan sehingga anak dapat mudah menerima hal yang baru dan menarik. Di usia seperti ini juga waktu yang tepat untuk mengajarkan kepada anak mengenai benda-benda atau objek yang ada di sekitarnya sehingga dapat membantu perkembangan kemampuan kognitifnya (Priyono dkk., 2021; Yuni Wulandari dkk., 2022).

Pengenalan objek adalah salah satu bidang di dalam bidang *computer vision*, yang digunakan untuk mengenali objek yang ada di sekitar dan bertujuan untuk mengidentifikasi dan melokalisasi objek-objek tertentu dalam gambar atau video. Pengenalan objek semakin banyak diminati sejak 1960-an dan terus berkembang hingga saat ini. Hal ini dibuktikan dengan penerapan *object detection* di berbagai bidang seperti di bidang medis dan bidang industri (Aini dkk., 2021; Karlina & Indarti, 2019; Rafly Alwanda dkk., 2020; Wu dkk., 2020). Pengenalan objek telah menjadi elemen kunci dalam berbagai aplikasi teknologi yang melibatkan analisis visual dan pemahaman konteks (Karlina & Indarti, 2019; Rafly Alwanda dkk., 2020). Dalam beberapa dekade terakhir, kemajuan dalam teknologi pengolahan

citra dan *computer vision* telah memungkinkan pengenalan objek menjadi lebih efisien dan akurat.

You Only Look Once (YOLO) adalah salah satu arsitektur dalam bidang *computer vision* yang bisa digunakan untuk pengenalan objek. YOLO menggunakan pendekatan yang menggabungkan deteksi dan klasifikasi objek dalam satu tahap (*one-stage-detector*) sehingga memberikan kecepatan dan efisiensi yang signifikan dibandingkan dengan metode tradisional yang memerlukan beberapa tahap pemrosesan (*two-stage-detector*) (Karlina & Indarti, 2019). YOLO telah menjadi salah satu pendekatan yang populer dalam pengenalan objek *real-time* dan banyak penelitian terkait telah memperluas aplikasi teknik ini dalam berbagai konteks seperti yang dilakukan dalam beberapa penelitian terakhir ini (Adarsh & Rathi, 2020; Aini dkk., 2021; Karlina & Indarti, 2019; Kumari dkk., 2021; Lou dkk., 2023; Zhang dkk., 2019).

Dengan memanfaatkan kecepatan dan efisiensi YOLO dalam mengenali objek secara *real-time* (Redmon dkk., 2015), penelitian ini bertujuan untuk menciptakan sebuah metode pembelajaran yang dapat membantu anak-anak dalam memahami atau mengenali lingkungan sekitar dengan mendekripsi objek-objek tersebut. Diharapkan bahwa integrasi teknologi ini akan memberikan kontribusi positif dalam memfasilitasi proses belajar anak-anak pada usia dini, memperluas cakupan pemahaman mereka terhadap objek di sekitar, dan mendukung perkembangan kemampuan kognitif serta pemahaman visual mereka secara menyeluruh.

1.2. Rumusan Masalah

Usia dini merupakan masa-masa emas untuk anak-anak dalam mempelajari objek-objek yang ada di sekitar untuk membantu perkembangan kemampuan kognitif anak-anak. Dalam beberapa hal, teknologi belum dapat dimanfaatkan secara optimal untuk membantu perkembangan kemampuan kognitif anak-anak. Karena itu, bagaimana penerapan teknologi *computer vision* khususnya menggunakan YOLO agar dapat membantu orang tua dalam mengajarkan kepada anak-anak dalam pengenalan objek sekitar?

1.3. Batasan Masalah

Pada penelitian ini terdapat beberapa batasan, yaitu:

1. Usia anak-anak yang menjadi fokus penelitian ini adalah anak-anak *preschool* dengan rentang umur 2 sampai 4 tahun.
2. Pada pengenalan objek untuk anak-anak, peneliti hanya berfokus pada objek-objek yang ada di sekitar dan mudah ditemui.
3. Objek pengenalan yang dijadikan *dataset* akan berbentuk gambar yang memuat 6 objek yang berbeda, yaitu: *handphone*, mobil, tas, manusia, jam, dan sepatu.
4. Kuesioner yang dilakukan akan mengacu pada aspek kegunaan sehingga yang menjadi target kuesioner adalah guru kelompok bermain, orang tua, masyarakat umum.

1.4. Tujuan Penelitian

Adapun tujuan penelitian ini adalah:

1. Menerapkan atau membuat sistem pengenalan objek menggunakan arsitektur YOLO yang dapat membantu anak-anak dalam mempelajari

objek-objek yang ada di sekitarnya dengan lebih mudah dan efektif sehingga dapat meningkatkan kualitas anak.

2. Mengukur dan mengevaluasi hasil penelitian dalam pengenalan objek sekitar berdasarkan *confusion matrix* dan mAP untuk melihat kebagusan model.
3. Mengukur dan mengevaluasi hasil penelitian berdasarkan kuesioner untuk melihat aspek kegunaan *website*.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Meningkatkan pembelajaran anak pada usia dini. Penelitian ini dapat membantu meningkatkan kualitas pembelajaran anak-anak usia dini dengan memanfaatkan teknologi *computer vision*. Anak-anak akan dapat belajar lebih efektif dengan bantuan pengenalan objek yang lebih baik.
2. Pengembangan lingkungan pendidikan. Hasil penelitian ini dapat membantu dalam pengembangan lingkungan pendidikan yang lebih modern dan adaptif, yang dapat memenuhi tuntutan era digital.
3. Kontribusi pada penelitian selanjutnya. Penelitian ini dapat menjadi dasar untuk penelitian lebih lanjut dalam penggunaan teknologi *computer vision* dalam konteks pendidikan anak-anak usia dini, dan dapat memberikan panduan bagi pengembangan teknologi serupa.

1.6. Sistematika Penulisan

Dalam penulisan penelitian ini terdapat lima bab utama. Pada bab I akan berbicara mengenai latar belakang penulisan, rumusan masalah, batasan masalah, tujuan penelitian, dan manfaat penelitian. Setelah membahas mengenai latar

belakang, maka pada bab II akan memuat teori-teori atau konsep-konsep yang akan dipakai pada penelitian ini. Dan pada bab III akan berisi tentang metodologi yang akan digunakan di dalam penelitian ini. Selanjutnya pada bab IV akan berisi pembahasan dan analisis dari langkah-langkah penelitian yang dijelaskan pada bab III. Dan yang terakhir pada bab V akan berisi kesimpulan terkait penelitian yang sudah dilakukan dan saran terhadap pengembangan selanjutnya.

BAB II:

TINJAUAN PUSTAKA

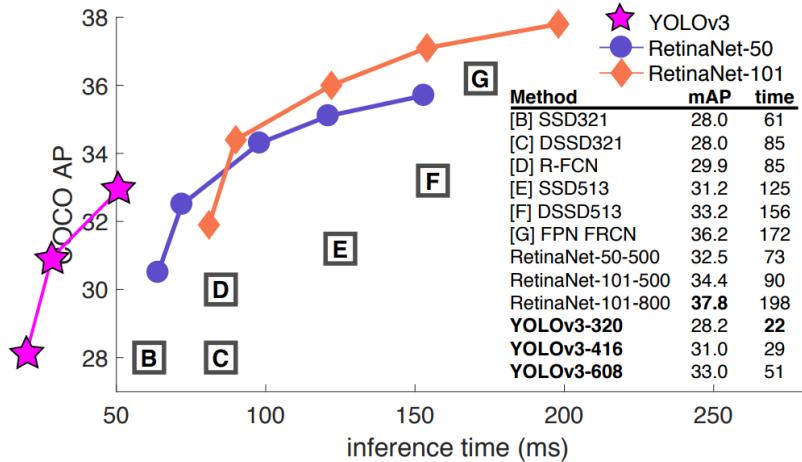
2.1. Review Literature

Pada penelitian sebelumnya telah berhasil di lakukan pengenalan objek binatang berbasis *Augmented Reality* (AR) oleh I Dewa Gede Wahya Dhiyatmika dan teman-temannya (Dhiyatmika dkk., 2015). Dalam perjalanan seturut perkembangan *deep learning* juga telah banyak dilakukan penelitian untuk mendeteksi objek seperti yang dilakukan oleh Muhammad Fadhlwan Supriadi dan teman-teman (Supriadi dkk., 2021) menggunakan *MobileNet*. Penelitian ini berhasil mendapatkan *mean Average Precision* (mAP) sebesar 99,34%. Pada penelitian yang dilakukan Ike Yuni Wulandari dan teman-teman (Yuni Wulandari dkk., 2022) pun mendapatkan hasil yang baik dan efektif. Pada penelitian ini pun disimpulkan bahwa dengan menggunakan *python* objek dapat dikenali dengan baik dan variasi pengambilan data tidak menjadi hambatan.

Selain menggunakan dukungan AR dan Raspberry, algoritma *Convolutional Neural Network* (CNN) dapat digunakan untuk membantu pengenalan objek seperti yang dilakukan oleh Dennis Saputra Ariansyah (Ariansyah, t.t.). Pada penelitian ini Dennis mencoba mengklasifikasi hewan dengan menggunakan CNN dan *transfer learning* dari *GoogleNet*. Penggunaan CNN dan *transfer learning* ini cukup baik dalam klasifikasi hewan yang dibuktikan dengan tingkat akurasi mencapai 98,36%. Ada lagi penelitian untuk pengenalan gestur jari tangan yang dilakukan Muhammad Rifki Pratama dan teman-teman (Heri Pratikno dkk., 2023). Penelitian ini mencoba dua cara dalam pengenalan gestur jari, yaitu *framework MediaPipe* dan CNN. Dari

penelitian ini diketahui bahwa penggunaan CNN kurang optimal dibandingkan *MediaPipe* jika dilakukan secara *real-time*.

Berbicara mengenai pengenalan objek secara *real-time*, salah satu algoritma yang menonjol dalam deteksi objek adalah YOLO (*You Only Look Once*). Beberapa studi telah menunjukkan keunggulan YOLO dibandingkan algoritma CNN dalam hal kecepatan. Menurut penelitian yang dilakukan oleh Pranav Adarsh, Pratibha Rathi, dan Manoj Kumar (Adarsh & Rathi, 2020), YOLO mampu mendekripsi objek dalam waktu yang lebih singkat dibandingkan dengan metode CNN, sembari tetap memberikan hasil deteksi yang baik. Hal ini pun didukung dengan hasil yang diteliti oleh Joseph Redmon (Redmon & Farhadi, 2018) yang menunjukkan kecepatan deteksi oleh YOLOv3 ketika dibandingkan dengan beberapa arsitektur deteksi seperti yang dapat dilihat pada **Gambar 2. 1.**



Gambar 2. 1: Grafik perbandingan beberapa arsitektur berdasarkan mAP dan waktu

Pada grafik tersebut dapat dilihat bahwa YOLO memiliki waktu yang lebih singkat dibandingkan dengan RetinaNet-50 atau RetinaNet-101 yang merupakan salah satu arsitektur dari CNN. Walaupun semakin besar ukuran gambar maka semakin meningkat waktu yang dibutuhkan, hal ini tetap membuktikan bahwa YOLO lebih cepat dibandingkan dengan CNN di mana pada kasus pada grafik YOLO

membutuhkan waktu 51 detik gambar dengan ukuran 608 sedangkan RetinaNet-50 membutuhkan waktu 73 detik untuk mendeteksi gambar dengan ukuran 500 atau RetinaNet-101 yang membutuhkan waktu 90 detik untuk mendeteksi gambar dengan ukuran 500.

Kecepatan ini dapat dicapai karena arsitektur YOLO menggunakan pendekatan satu jaringan tunggal untuk deteksi objek. Sebaliknya, metode CNN biasanya melibatkan proses deteksi yang lebih kompleks dan berlapis-lapis. Penelitian lainnya oleh Haitong Lou dan rekan-rekannya (Lou dkk., 2023) juga menunjukkan bahwa YOLO memiliki performa yang unggul dalam deteksi objek baik dalam situasi *real-time* maupun *non-real-time*. Lebih lanjut dari perbedaan YOLO dan CNN pun dapat dilihat pada poin **2.2.5** yang di dalamnya akan membahas terkait perbedaan antara CNN dan YOLO.

Secara lebih lanjut berikut adalah tabel *review literatur* mengenai penelitian-penelitian yang berkaitan dengan penelitian yang akan dibuat.

Tabel 2. 1: Tabel review literature

No.	Judul, Peneliti, Penerbit dan Tahun Terbit	Hasil Penelitian, Keterbatasan, Peluang
1.	Judul: Pengenalan Gestur Jari Tangan Sebagai Media Pembelajaran Berhitung Bagi Paud Berbasis Visi Komputer dan Deep Learning	Hasil Penelitian: Hasil penelitian ini menunjukkan bahwa dalam konteks pengenalan gestur jari tangan sebagai media pembelajaran berhitung bagi anak paud, arsitektur <i>Convolutional Neural Network</i> (CNN) mencapai akurasi hasil pelatihan 100% pada <i>epoch</i> ke-5, dengan total waktu komputasi 17,113 detik. CNN memerlukan waktu komputasi 12 detik pada setiap langkah (step) dan 3,366 - 3,452 detik pada setiap <i>epoch</i> -nya. Namun, saat dibandingkan dengan <i>MediaPipe</i> , hasil komparasi menunjukkan bahwa <i>MediaPipe</i> memiliki persentase akurasi rata-rata sebesar 89,9% dengan FPS (<i>frame per second</i>) antara 25-30, sedangkan CNN memiliki persentase akurasi

		rata-rata sebesar 20% dengan FPS antara 12-15. Performa CNN kurang optimal untuk deteksi objek secara <i>real-time</i> karena memerlukan proses pelatihan <i>dataset</i> gestur jari tangan yang membebani komputasi, sedangkan <i>MediaPipe</i> dirancang khusus untuk deteksi gestur tangan.
	Peneliti: Muhammad Rifki Pratama, Heri Pratikno, Yosefine Triwidystuti, dan Musayyanah	Keterbatasan: Kelemahan penelitian ini adalah performa CNN dalam deteksi objek <i>real-time</i> ternyata kurang optimal, dan metode ini lebih sesuai untuk klasifikasi citra. Selain itu, hasil pelatihan CNN mencapai akurasi 100%, tetapi performanya dalam proses pengujian (testing) tidak sebaik <i>MediaPipe</i> , yang memiliki akurasi yang lebih tinggi dan FPS yang lebih tinggi.
	Penerbit dan Tahun Terbit: COMPLETE: Journal of Computer, Electronic, and Telecommunication, 2023	Peluang: Peluang dari penelitian ini adalah pengembangan lebih lanjut dengan membandingkan kinerjanya dengan arsitektur <i>deep learning</i> lainnya, seperti menggunakan <i>pre-trained network</i> model seperti LSTM, Faster-RCNN, dan ResNet5.0. Hal ini dapat memberikan wawasan lebih lanjut tentang metode terbaik untuk deteksi gestur jari tangan dalam konteks pembelajaran berhitung anak paud. Selain itu, pengembangan metode yang lebih efisien untuk CNN dalam deteksi objek <i>real-time</i> mungkin dapat meningkatkan kinerjanya, sehingga dapat menjadi lebih bersaing dengan <i>MediaPipe</i> dalam hal akurasi dan FPS.
2.	Judul: You Only Look Once: Unified, Real-Time Object Detection	Hasil Penelitian: Penelitian ini memperkenalkan YOLO, sebuah model yang menyatukan pendekatan untuk deteksi objek. Model ini dirancang dengan sederhana dan dapat dilatih langsung pada gambar utuh. YOLO juga diakui sebagai pendekatan tercepat dalam literatur untuk deteksi objek secara umum dan mendorong perkembangan dalam deteksi objek <i>real-time</i> . Selain itu, YOLO mampu memberikan hasil yang baik dalam berbagai domain, menjadikannya ideal untuk aplikasi yang

		mengandalkan deteksi objek yang cepat dan andal.
	Peneliti: Joseph Redmon, Santosh Divvala, Ross Girshick, dan Ali Farhadi	Keterbatasan: Berdasarkan jurnal dan hasil penelitian, tidak ditemukan keterbatasan karena berhubung ini adalah sebuah penemuan yang menjadi titik tolak kemajuan algoritma YOLO
	Penerbit dan Tahun Terbit: Proceedings of the IEEE conference on computer vision and pattern recognition 2015	Peluang: Dikarenakan tidak adanya keterbatasan yang diberikan, kemungkinan peluang yang dapat diambil dari penelitian ini adalah mengenai pengaplikasian YOLO secara <i>real-time</i> dan pengaplikasian ke dalam berbagai hal.
3.	Judul: YOLBO: You Only Look Back Once—A Low Latency Object Tracker Based on YOLO and Optical Flow	Hasil Penelitian: Penelitian ini membangun pada paradigma pelacakan yang sudah berhasil dan dapat diterapkan pada berbagai kasus penggunaan dan tipologi perangkat keras yang berbeda.
	Peneliti: Daniel S. Kaputa, dan Brian P. Landy	Keterbatasan:
	Penerbit dan Tahun Terbit: IEEE Access 2021	Peluang: Peluang yang bisa dikembangkan dari penelitian ini berkaitan dengan penelitian yang menggunakan <i>frame rate</i> dinamis, dan penggunaan YOLBO pada embedded DPGA Soc
4.	Judul: A Real-Time Method to Estimate Speed of Object Based on Object Detection and Optical Flow Calculation	Hasil Penelitian: Dari penelitian ini didapatkan bahwa metode yang digunakan dapat mengestimasi kecepatan dari berbagai jenis objek.
	Peneliti: Kaizhan Liu, Yunming Ye, Xutao Li, dan Yan Li	Keterbatasan:
	Penerbit dan Tahun Terbit: Journal of Physics: Conference Series 2018	Peluang: Peluang yang dapat diambil dari penelitian ini adalah mengenai pengaplikasian di berbagai bidang yang memerlukan pengukuran kecepatan objek, juga dapat dikembangkan untuk mengukur atau menganalisis sebuah pergerakan.

5.	Judul: Deteksi Masker Wajah Menggunakan Deep Transfer Learning dan Augmentasi Gambar	Hasil Penelitian: Hasil penelitian menunjukkan bahwa penggunaan teknik <i>deep transfer learning</i> dan augmentasi gambar dapat meningkatkan kinerja model deteksi masker wajah secara signifikan. Penerapan kedua pendekatan tersebut memberikan kontribusi peningkatan kinerja sebesar 12-13%. Berdasarkan pengujian akhir model <i>deep learning</i> yang dibangun mencapai akurasi 98,3% dan skor F1 98,7% pada <i>dataset</i> validasi. Penelitian selanjutnya dapat diarahkan untuk mendeteksi hal yang lebih kompleks seperti mendeteksi penggunaan masker ganda (masker bedah dan masker kain).
	Peneliti: Raden B. Hadiprakoso, dan Nurul Qomariasih	Keterbatasan:
	Penerbit dan Tahun Terbit: Jurnal Informatika dan Komputer 2022	Peluang: Peluang yang dapat dilanjutkan dari penelitian ini mengenai penerapan <i>deep learning</i> dan augmentasi dalam mendeteksi objek di berbagai bidang.
6.	Judul: Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO)	Hasil Penelitian: Dari penelitian ini menghasilkan model penerjemah alfabet BISINDO dengan algoritma CNN yang mencapai akurasi 94.38% padahal sebelumnya tanpa menggunakan <i>pre-processing</i> hanya mencapai akurasi 30%.
	Peneliti: Riestiya Zain Fadillah, Ade Irawan, dan Meredita Susanty	Keterbatasan: Keterbatasan yang terdapat dari penelitian ini berupa ruang lingkup terjemahan yaitu alfabet.
	Penerbit dan Tahun Terbit: Jurnal Informatika 2021	Peluang: Peluang yang dapat dilanjutkan adalah mengenai perluasan cakupan penerjemahan hingga pada penerjemahan kata dan kalimat.
7.	Judul: A Review of YOLO Algorithm Developments	Hasil Penelitian: Hasil penelitian ini mengidentifikasi bahwa YOLO versi-versi memiliki banyak perbedaan, tetapi masih memiliki beberapa fitur yang sama, sehingga tetap memiliki kemiripan. Ini menunjukkan bahwa ada ruang bagi penelitian lebih lanjut dalam pengembangan versi-versi

		YOLO, terutama dalam konteks implementasi skenario.
	Peneliti: Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, dan Bo Ma	Keterbatasan: Keterbatasan makalah ini termasuk kurangnya fokus pada implementasi perbandingan, seperti analisis skenario, yang bisa menjadi area penelitian yang lebih mendalam.
	Penerbit dan Tahun Terbit: Procedia Computer Science 2022	Peluang: Peluang yang dapat dilanjutkan dari penelitian ini mengenai implementasi YOLO.
8.	Judul: YOLO v3-Tiny: Object Detection and Recognition using one stage improved model	Hasil Penelitian: Dari penelitian ini ditemukan bahwa YOLO v3-Tiny dapat meningkatkan kecepatan deteksi objek dengan akurasi yang bagus.
	Peneliti: Pranav Adarsh, Pratibha Rathi, dan Manoj Kumar	Keterbatasan: Keterbatasan yang ada di dalam penelitian ini terkait penanganan pencahayaan yang kurang bagus, variasi skala objek dan kompleksitas latar belakang.
	Penerbit dan Tahun Terbit: International Conference on Advanced Computing & Communication Systems (ICACCS) 2020	Peluang: Peluang yang ada di dalam penelitian ini berkaitan dengan meningkatkan akurasi lokalisasi target yang kecil, juga memperluas deteksi objek gambar statis.
9.	Judul: DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor	Hasil Penelitian: Hasil penelitian ini adalah pengembangan algoritma deteksi objek berukuran kecil yang berbasis pada sensor kamera yang dikombinasikan dengan kecerdasan buatan (<i>artificial intelligence</i>). Algoritma ini mencapai hasil yang lebih baik daripada algoritma-algoritma sebelumnya dalam hal akurasi dan kecepatan deteksi. Eksperimen dan pengujian dilakukan pada beberapa dataset, termasuk Visdrone, TinyPerson, dan PASCAL VOC2007, dan algoritma ini berhasil membuktikan keunggulannya dalam mendeteksi objek kecil dalam berbagai situasi kompleks.
	Penulis: Haitong Lou, Xuehu	Keterbatasan:

	Duan, Junmei Guo, Haiying Liu, Jason Gu, Lingyun Bi, dan Haonan Chen	Keterbatasan umum dalam penelitian deteksi objek mungkin termasuk performa yang kurang baik dalam situasi pencahayaan yang buruk, kebingungan dalam mengenali objek yang tumpang tindih, atau pengaruh latar belakang yang rumit.
	Penerbit dan Tahun Terbit: Electronics 2023	Peluang: Peluang yang dapat dilanjutkan atau dikembangkan adalah mengenai peningkatan algoritma deteksi objek untuk berbagai ukuran objek. Penelitian tersebut ingin mencapai tingkat akurasi yang lebih baik dalam waktu secepat mungkin.

2.2. Landasan Teori

2.2.1. Pendidikan Anak Usia Dini

Anak usia dini, yaitu anak dengan rentang usia 0 hingga 6 tahun, merupakan sosok sosiokultural yang sedang mengalami perkembangan fundamental. Pada fase ini, anak mampu menerima dan mengolah informasi dengan cepat serta tahan lama. Anak-anak akan memasuki tahap pra-operasional antara usia 2 hingga 7 tahun, di mana mereka mulai menggunakan citra-citra untuk mengenali lingkungan sekitar serta mengembangkan kemampuan berpikir simbolis. (Putri dkk., 2021; Safita & Suryana, 2022; Sunarti dkk., 2023; Zulwati dkk., 2022).

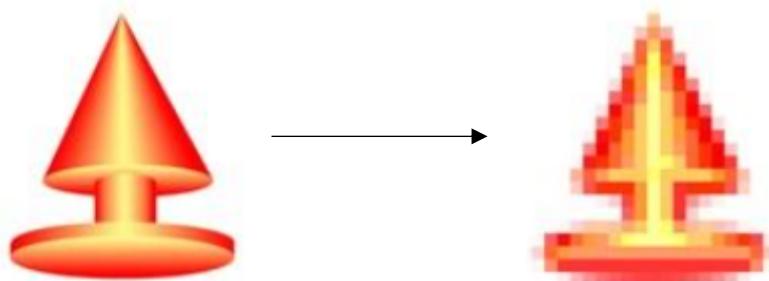
Pendidikan usia dini, selain menekankan nilai agama dan moral, juga memiliki fokus pada pengembangan kognitif anak sebagai dasar pijakannya di masa yang akan datang. Perkembangan kognitif anak usia dini merupakan tahap penting dalam proses belajar dan memahami dunia di sekitarnya. Karena itu, stimulus yang tepat dapat membantu perkembangan ini berjalan optimal (Nur dkk., 2020). Aspek kognitif menjadi hal utama dalam perkembangan anak karena berkaitan dengan bagaimana cara mereka berpikir dan mulai berfungsi. Karena itu, tujuan utama

pendidikan anak usia dini dalam hal perkembangan kognitif adalah agar anak mampu berpikir kritis, menalar, memecahkan masalah, dan menemukan sebab-akibat dari hasil belajarnya secara mandiri (Komang Ayu & Surya Manuaba, 2021; Nur dkk., 2020).

Kelompok bermain atau PAUD dirancang khusus untuk merangsang perkembangan kognitif anak usia yang memasuki usia 2 – 4 tahun dengan melalui berbagai kegiatan bermain yang edukatif dan menyenangkan. Dengan stimulasi dan pendidikan yang tepat, anak usia 2 – 4 tahun akan memiliki fondasi kognitif yang kuat yang akan bermanfaat bagi perkembangan mereka selanjutnya (Putri dkk., 2021).

2.2.2. Image Processing

Citra adalah suatu gambaran atau kemiripan atau suatu imitasi dari suatu objek (Andono dkk., 2017; Hidayatullah, 2017). Pada umumnya citra dibagi menjadi dua jenis, yaitu citra analog dan citra digital (Iryanto & Zaini, 2014). Citra analog tidak dapat direpresentasikan oleh komputer. Agar komputer dapat mengolah citra analog tersebut, citra harus melewati proses sampling dan kuantisasi yang disebut “digitalisasi” di mana proses digitalisasi ini akan mengubah citra analog menjadi citra digital (Andono dkk., 2017). Proses digitalisasi ini akan memecahkan citra analog menjadi sejumlah baris (m) dan kolom (n). Sehingga citra digital $a [m, n]$ dapat dijelaskan sebagai ruang diskrit 2D yang berasal dari sebuah citra analog $a [x, y]$. Titik pertemuan antara baris dan kolom dalam citra digital disebut *pixel* (Andono dkk., 2017; Iryanto & Zaini, 2014).

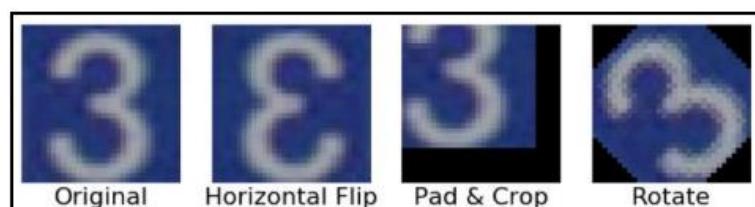


Gambar 2. 2: Ilustrasi proses digitalisasi

Sumber: <https://slideplayer.info/slide/11809302/>

Citra pada umumnya menyimpan banyak informasi. Namun, seringnya citra mengalami penurunan kualitas citra sehingga dibutuhkan sebuah pemrosesan citra untuk meningkatkan kualitasnya (Adhinata dkk., 2020; Hidayatullah, 2017; Panggali dkk., 2022). Pemrosesan citra adalah bidang studi yang fokus pada pengolahan dan analisis data pada citra digital yang bertujuan untuk meningkatkan kualitas citra. Tahapan pengolahan citra digital pada umumnya adalah akuisisi citra, peningkatan kualitas citra, segmentasi citra, ekstraksi fitur citra, dan klasifikasi citra (Adhinata dkk., 2020). Hasil dari pengolahan citra sebagian besar dapat mempengaruhi bagian tingkat tinggi selanjutnya untuk melakukan pengenalan dan pemahaman terhadap data citra (Chen dkk., 2021).

2.2.3. Augmentasi



Gambar 2. 3: Ilustrasi proses augmentasi

Sumber: <https://www.51cto.com/article/626105.html>

Augmentasi data adalah teknik untuk memperluas *dataset* pelatihan dengan membuat variasi pada data yang ada dengan memanipulasi transformasi dimensi gambar (Perez dkk., 2018; Sanjaya & Ayub, 2020). Teknik augmentasi data seperti

cropping, *padding*, dan *flipping horizontal* bertujuan untuk meningkatkan kinerja model pembelajaran mesin dengan memberikan lebih banyak variasi pada data pelatihan, sehingga model dapat belajar fitur yang lebih umum dan dapat digeneralisasi dengan baik pada data baru. Selain itu, dengan augmentasi pun dapat mengurangi *overfitting* seminimal mungkin. (Fadillah dkk., 2021; Perez dkk., 2018).

2.2.4. Computer Vision

Computer vision adalah cabang ilmu komputer yang berfokus pada pengembangan algoritma dan teknik untuk memungkinkan komputer memahami dan menafsirkan informasi dari gambar digital atau video. *Computer vision* akan melibatkan berbagai disiplin ilmu seperti pengolahan citra, pembelajaran mesin, kecerdasan buatan, dan *deep learning*. Salah satu bidang yang ada pada *computer vision* adalah deteksi dan pengenalan objek (*object detection and recognition*), yang melibatkan identifikasi dan klasifikasi beberapa objek dalam gambar yang sering digunakan dalam aplikasi seperti sistem pengawasan keamanan dan analisis citra medis (Nalbant & Uyanik, 2021).

2.2.5. You Only Look Once (YOLO)

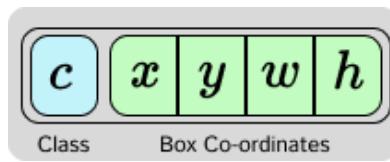
You Only Look Once (YOLO) adalah sebuah arsitektur yang menerapkan pendekatan baru untuk mendeteksi objek. YOLO pertama kali dikenalkan pada tahun 2015 (Redmon dkk., 2015). Algoritma ini biasanya digunakan untuk mendeteksi objek secara *realtime*. YOLO menggunakan jaringan saraf tiruan untuk mempelajari pola-pola sehingga dapat mendeteksi dan mengenali objek sekaligus memprediksi *bounding box* dan kelasnya. YOLO merupakan algoritma *One-stage Detector* hasil dari pengembangan algoritma *Convolutional Neural Network* (CNN)

yang menggabungkan proses ekstraksi fitur dan proses regresi atau klasterisasi dalam satu *network* (Thoriq dkk., 2023). Hal inilah yang membuat YOLO dapat memprediksi *bounding box* dan probabilitas kelas langsung dari gambar penuh dalam satu evaluasi (Jiang dkk., 2022; Redmon dkk., 2015).

Walaupun YOLO dikembangkan dari algoritma CNN, kedua algoritma ini memiliki beberapa perbedaan yang signifikan. YOLO memiliki arsitektur yang lebih sederhana dan efisien dengan menggabungkan beberapa tahap dalam proses deteksi objek menjadi satu langkah tunggal, yang membuatnya sangat cepat. Di sisi lain, CNN biasanya terdiri dari beberapa tahap terpisah seperti ekstraksi fitur, proposal region, dan klasifikasi, yang menjadikannya lebih kompleks dan memerlukan lebih banyak waktu pemrosesan.

Dari sisi kecepatan prediksi, YOLO unggul karena mampu memproses gambar dalam waktu nyata (*real-time*) dengan kecepatan yang sangat tinggi, sehingga sangat cocok untuk aplikasi yang memerlukan deteksi objek secara langsung. Sementara itu, CNN sering digunakan untuk aplikasi yang tidak memerlukan kecepatan real-time, namun lebih fokus pada akurasi tinggi dan kemampuan mendeteksi objek-objek yang lebih kecil atau lebih rumit, meskipun dengan waktu pemrosesan yang lebih lama (Kulsum & Cherid, 2023; A. Maulana dkk., 2024; Putra dkk., 2016).

Selain itu, kecepatan prediksi yang dilakukan oleh YOLO juga dipengaruhi oleh penggunaan *grid* di mana setiap *grid* tersebut bertanggung jawab dalam memprediksi objek di dalamnya, sehingga dapat mengurangi wilayah yang perlu diproses secara individual. (Alberto, Joseph ; Hermanto, 2023; Gajalakshmi dkk., 2020; A. Maulana dkk., 2024).



Gambar 2. 4: Ilustrasi nilai yang merepresentasikan bounding box

Setiap *bounding box* yang dihasilkan dan/atau dibuat pada YOLO akan menghasilkan lima nilai di dalamnya. Nilai-nilai ini terdiri dari nilai *c*, nilai *x* dan *y*, nilai *w* dan *h*, di mana *x* dan *y* mewakili pusat dari kotak relatif ke batas sel *grid*, *w* dan *h* merupakan lebar dan tinggi dari *bounding box*, dan *c* adalah jenis objek yang ada. Hal inilah yang membuat YOLO menjadi algoritma yang cepat dan akurat (Kaputa & Landy, 2021; Kumari dkk., 2021; Lou dkk., 2023).

Ultralytics merilis YOLO v8 yang dikembangkan dari versi-versi sebelumnya. Pada YOLO v8 ini terdapat beberapa perubahan pada arsitektur dan optimasi terhadap akurasinya (Jocher & Sergiuwaxmann, 2023). Secara garis besar, arsitektur YOLO v8 terdiri dari tiga bagian utama: *backbone*, *neck*, dan *head* (Lou dkk., 2023).

Backbone bertugas sebagai penyaring fitur dasar dari gambar yang masuk. Pada YOLOv8, *backbone* atau tulang punggung menggunakan CSPDarknet53 yang dimodifikasi. Ditahap ini, data *input* akan melewati lapisan konvolusi yang membantu mengekstraksi fitur-fitur lokal seperti tepi, sudut, dan tekstur, kemudian melewati lapisan *pooling* yang akan mengurangi dimensi spasial dari fitur-fitur tersebut. *Neck* yang berada di antara *backbone* dan *head* akan menyusun kembali fitur-fitur yang telah diekstrak oleh *backbone*. Penyusunan kembali ini akan melibatkan beberapa lapisan konvolusi dan *pooling* tambahan sehingga dapat membantu *neck* untuk menyusun kembali informasi yang relevan. Bagian terakhir adalah *head*. *Head* akan bertugas untuk mengklasifikasi, regresi, dan mendeteksi

objek berdasarkan fitur-fitur yang telah disusun oleh *neck*. *Neck* bertujuan untuk mengintegrasikan informasi yang didapatkan sehingga menghasilkan representasi fitur yang lebih kompleks dan kuat (Jocher & Sergiuwaxmann, 2023; Khare dkk., 2023; Lou dkk., 2023).

2.2.6. Hyperparameter YOLOv8

Untuk meningkatkan performa dari model yang dibuat, dapat dilakukan dengan mencari kombinasi *hyperparameter* YOLOv8. Adapun *hyperparameter* yang bisa digunakan adalah:

2.2.6.1. Pre-Trained Model

Pre-trained model merupakan jenis pembelajaran transfer di mana pengetahuan dari berbagai tugas sumber diambil dan diterapkan pada tugas target. Tujuannya adalah untuk menggunakan pengetahuan yang telah dipelajari sebelumnya untuk memecahkan masalah baru dan mencapai hasil yang lebih baik. Metode *pre-trained* ini menjembatani tugas sumber dan tugas target, mengkodekan pengetahuan dari tugas sumber dan mentransfernya ke tugas target (Han dkk., 2021).

2.2.6.2. Dropout

Dropout adalah teknik regulasi yang digunakan untuk mencegah *overfitting* dalam jaringan saraf. Selama pelatihan, *dropout* secara acak menonaktifkan sejumlah neuron dalam jaringan untuk setiap iterasi, yang memaksa jaringan untuk tidak bergantung pada neuron tertentu dan mendorong pembelajaran yang lebih umum. Ini membantu meningkatkan kemampuan jaringan untuk menggeneralisasi dari data pelatihan ke data yang belum pernah dilihat sebelumnya (Wei dkk., 2020).

2.2.6.3. Batch

Batch mengacu pada sekelompok data pelatihan yang digunakan untuk satu iterasi pembaruan bobot dalam jaringan saraf. Penggunaan *batch* akan memperbarui bobot setelah memproses beberapa contoh data sekaligus. Pendekatan ini menyeimbangkan efisiensi komputasi dan stabilitas pembaruan bobot, membuat pelatihan lebih cepat dan stabil (Karna dkk., 2023; Singh dkk., 2020).

2.2.6.4. Learning Rate

Learning rate adalah parameter yang menentukan seberapa besar pembaruan bobot dilakukan pada setiap iterasi pelatihan. *Learning rate* yang terlalu tinggi dapat menyebabkan pelatihan menjadi tidak stabil dan gagal konvergen, sementara *learning rate* yang terlalu rendah dapat membuat pelatihan sangat lambat dan mungkin terjebak dalam minima lokal (Wen dkk., 2021).

2.2.6.5. Optimizer Adam

Adam merupakan singkatan dari *Adaptive Moment Estimation*. Adam menggabungkan keuntungan dari dua teknik optimasi yang berbeda yaitu momentum dan adaptif *learning rate*. Dengan menggunakan rata-rata momentum dari gradien sebelumnya untuk mempercepat pelatihan, Adam mengurangi fluktuasi yang dapat menghambat konvergensi. Selain itu, Adam menyesuaikan *learning rate* untuk setiap parameter secara individual berdasarkan estimasi pertama dan kedua momen dari gradien, sehingga dapat menyesuaikan diri dengan perubahan dinamis dalam gradien. Ini membuat Adam sangat efisien dan cepat dalam konvergensi, serta mampu menangani masalah dengan data yang *noise* atau gradien yang jarang (Chandriah & Naraganahalli, 2021; Sandhya & Kashyap, 2024).

2.2.6.6. Optimizer RMSProp

RMSProp adalah kepanjangan dari *Root Mean Square Propagation*. RMSProp menyesuaikan *learning rate* secara adaptif untuk setiap parameter. RMSProp menghitung rata-rata eksponensial dari kuadrat gradien terbaru untuk menormalkan *learning rate*. Dengan cara ini, RMSProp dapat mengatasi masalah yang terkait dengan *gradient descent* standar di mana gradien yang besar dapat menyebabkan pembaruan yang tidak stabil. Ini memungkinkan RMSProp untuk lebih stabil dalam pelatihan, terutama dalam jaringan saraf yang mendalam dan pada masalah dengan gradien yang besar dan tidak stabil (Elshamy dkk., 2023; Sandhya & Kashyap, 2024).

2.2.7. Roboflow

Roboflow adalah sebuah platform perangkat lunak yang memiliki kemampuan dalam mengorganisasi data, anotasi sampai pada pelatihan dan penerapan model yang dibuat. Roboflow digunakan untuk membuat label *bounding box* pada dataset gambar yang bisa digunakan sebagai dataset pelatihan dan testing (Dwyer & Gallagher, 2023).

2.2.8. Labeling Gambar

Anotasi gambar atau pelabelan gambar merupakan sebuah proses memberikan label atau keterangan pada gambar. Label atau keterangan ini merupakan informasi lanjutan mengenai objek yang ada di dalamnya. Pelabelan ini biasanya berupa *bounding box* yang menunjukkan lokasi dari objek sekaligus mengidentifikasi dan mengategorikan objek dalam gambar. Proses ini sering dilakukan dalam konteks *computer vision* untuk melatih model yang mampu mengenali dan mengklasifikasi objek. Dengan pelabelan ini juga dapat membantu

model untuk mengenali pola objek dan membedakan tiap objek (Heri Pratikno dkk., 2023; F. Maulana, 2021).

2.2.9. Confusion Matrix

Confusion matrix adalah tabel yang digunakan untuk mengevaluasi hasil pemodelan pada klasifikasi *dataset*. *Confusion matrix* nantinya akan menampilkan atau memvisualisasikan gambaran lengkap mengenai hasil klasifikasi yang tepat dan yang salah. Pada umumnya terdapat empat nilai utama dalam *confusion matrix*, yaitu 1) *true positive*, 2) *true negative*, 3) *false negative*, dan 4) *false positive* (Maurya dkk., 2021).

2.2.10. Mean Average Precision

Mean Average Precision (mAP) adalah sebuah metrik evaluasi yang digunakan dalam bidang pemrosesan citra dan pengenalan objek, terutama dalam tugas deteksi objek. *Mean Average Precision* (mAP) mengukur sejauh mana model dapat mengidentifikasi objek dengan akurasi yang tinggi. *Mean Average Precision* (mAP) mengambil hasil dari *Precision-Recall* (P-R) *Curve* untuk setiap kelas objek dan kemudian menghitung *Average Precision* (AP) untuk setiap kelas berdasarkan *Area Under the Curve* (AUC) dari *P-R Curve* tersebut. Setelah itu, mAP mengambil rata-rata dari AP untuk semua kelas objek yang ada, sehingga semakin tinggi nilai mAP, semakin baik kinerja model deteksi objek tersebut (Maxwell dkk., 2021; Q. Wang dkk., 2018; Zhao & Li, 2020).

2.2.11. Skala Likert

Skala Likert yang dikembangkan oleh Rensis Likert merupakan sebuah skala pengukuran kemampuan yang bertujuan untuk melihat tingkat persetujuan terhadap pertanyaan atau pernyataan yang diberikan (Suasapha, 2020).

Kemampuan yang diukur ini akan dijabarkan menjadi beberapa indikator dan digunakan untuk menyusun instrumen. Bentuk kuesioner yang jamak ini adalah pernyataan yang disertai dengan skala pengukuran, di mana skala tersebut merupakan pilihan sikap terkait pernyataan yang mengikutinya. Karena itu, pilihan sikap yang sering kali dilihat pada kuesioner skala Likert biasanya mengenai persetujuan terhadap pernyataan atau pertanyaan terkait (Ayuka dkk., 2021; Suasapha, 2020).

Adapun jawaban instrumen dari skala Likert ini dikembangkan menjadi empat kategori, yaitu sangat setuju (SS), setuju (S), tidak setuju (TS), dan sangat tidak setuju (STJ) (Ayuka dkk., 2021).

2.2.12. Uji Validitas

Uji validitas adalah pengujian terhadap alat ukur untuk melihat sejauh mana sebuah instrumen dapat mengukur apa yang akan diukur. Uji validitas juga digunakan agar dapat menguji ketepatan dan ketetapan suatu alat ukur yang digunakan sebagai pengukur (Ayuka dkk., 2021; Rosita dkk., 2021). Alat ukur atau kuesioner dinyatakan valid jika setiap butir pertanyaan dapat digunakan sebagai perantara untuk mengungkapkan sesuatu yang akan diukur oleh kuesioner. Valid dan tidaknya sebuah alat ukur dilihat dari nilai rhitung yang lebih besar dari rtabel (Budiyanta, 2018; Rosita dkk., 2021).

Untuk menghitung valid tidaknya sebuah alat ukur dapat dilihat pada persamaan 2.

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \quad (1)$$

Di mana:

n = koefisien korelasi

Σx = jumlah skor item

Σy = jumlah skor total

n = jumlah responden

(Budiyanta, 2018; Ernawati & Sukardiyono, 2017).

2.2.13. Uji Reliabilitas

Uji reliabilitas digunakan untuk mengukur sejauh mana instrumen atau alat ukur memiliki kekonsistensi untuk mengukur apa yang semestinya diukur. Pengujian ini dilakukan untuk melihat sejauh mana alat ukur dapat dipercaya atau diandalkan. Alat ukur dikatakan reliabel jika menghasilkan hasil yang sama meskipun dilakukan pengukuran berkali-kali (Ernawati & Sukardiyono, 2017; Rosita dkk., 2021). Uji reliabilitas ini dilakukan setelah alat ukur melewati pengujian validitas dan dinyatakan valid. Pengukuran ini akan menggunakan metode Cronbach's Alpha yang menggunakan rumus seperti pada persamaan 3.

$$r = \left(\frac{k}{k-1} \right) \left(\frac{\sum \sigma_b^2}{\sigma_t^2} \right) \quad (2)$$

Di mana:

r = koefisien reliabilitas

k = jumlah butir pertanyaan sah

$\sum \sigma_b^2$ = jumlah varian butir

σ_t^2 = jumlah skor total

(Rosita dkk., 2021).

2.2.14. Uji Kegunaan (Usability)

Aspek *Usability* atau aspek kegunaan adalah aspek yang akan mengukur kemampuan produk sejauh mana produk dapat dipahami dan dipelajari oleh pengguna. *Usability* ini merupakan atribut yang berhubungan dengan upaya yang

diperlukan ketika pengguna mulai menggunakan perangkat yang dibuat atau dihasilkan. *Usability* menggambarkan seberapa puas pengguna dalam mengoperasikan perangkat. Ada empat sub aspek yang akan diliat dari sisi *usability*, yaitu: (1) sub *understandability* yang akan mengukur kemampuan perangkat untuk dapat dipahami dengan mudah; (2) Sub *learnability* yang akan mengukur kemampuan perangkat lunak untuk dapat dipelajari dengan mudah; (3) Sub *operability* yang akan mengukur kemampuan perangkat untuk dapat dioperasikan dengan mudah; dan (4) sub *attractiveness* yang akan mengukur kemampuan perangkat lunak untuk dapat tampil menarik bagi pengguna (Ernawati & Sukardiyono, 2017; Kusuma dkk., 2016; Sufandi dkk., 2022).

Untuk menghitung kegunaan dari sebuah perangkat berdasarkan kuesioner akan melalui rumus pada persamaan 4.

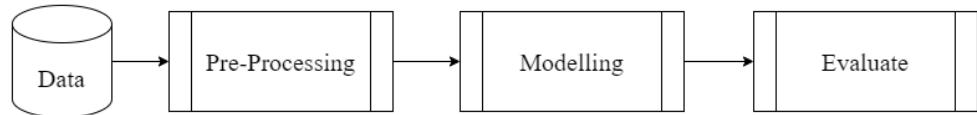
$$usability = \frac{skor\ yang\ diobservasi}{skor\ yang\ diharapkan} \times 100 \quad (3)$$

(Sufandi dkk., 2022).

BAB III

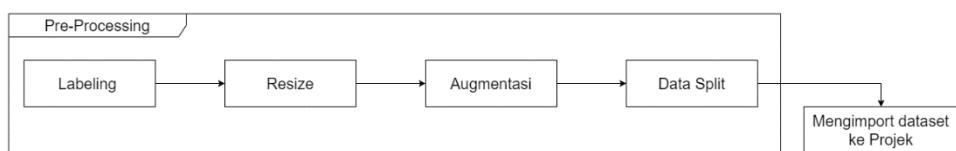
METODE PENELITIAN

3.1.Gambaran Umum Penelitian



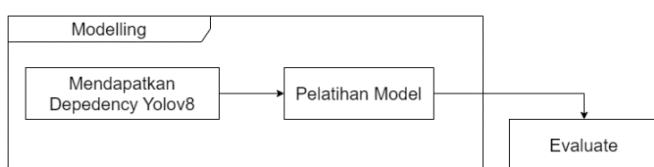
Gambar 3. 1: Flowchart penelitian

Secara garis besar, alur dari penelitian yang akan dilakukan menggunakan bahasa pemrograman *python* ini dapat dilihat dari **Gambar 3. 1**. Dari *flowchart* tersebut, terdapat empat tahap utama dalam penelitian ini. Pada penelitian ini setelah data dikumpulkan, selanjutnya akan dilakukan *pre-processing* sebelum dibuat model. Dan setelah model dibuat, akan dilakukan evaluasi terhadap model tersebut. Dalam tahap *pre-processing* dan modeling dapat di lihat secara rinci pada **Gambar 3. 2** dan **Gambar 3. 3**.



Gambar 3. 2: Flowchart pre-processing

Untuk pengolahan dataset dari data mentah akan menggunakan *tools* Roboflow. Hal ini dikarenakan Roboflow dapat membantu peneliti dalam mengolah data mulai dari pelabelan hingga pada *pre-processing* dan augmentasi dataset sehingga dapat mempermudah sekaligus mempercepat proses *pre-processing*.



Gambar 3. 3: Flowchart Modelling

Selanjutnya, detail dari keempat tahapan di atas dijelaskan pada poin-poin berikutnya.

3.2. Data



Gambar 3. 4: Gambar kumpulan dataset sesuai kelas

Dataset yang digunakan dalam penelitian ini adalah kumpulan gambar dari kelas-kelas objek seperti *handphone*, mobil, tas, manusia, jam, dan sepatu. Pemilihan objek-objek ini didasarkan pada fakta bahwa mereka sering ditemukan di sekitar kita, mempermudah proses eksplorasi dan pengenalan lingkungan sekitar.

Anak-anak pada usia ini sering kali terpapar pada objek-objek tersebut dalam kehidupan sehari-hari. Menurut Piaget (1952), anak-anak belajar lebih efektif ketika mereka dapat mengaitkan pembelajaran dengan hal-hal yang mereka lihat dan gunakan secara rutin (Piaget, 1952). Vygotsky (1978) juga menekankan pentingnya interaksi dengan objek nyata untuk mendukung perkembangan kognitif dan bahasa pada anak-anak (VYGOTSKY, 1978). Dengan demikian, objek-objek yang dipilih tidak hanya cocok untuk pengenalan objek bagi anak-anak usia 2-4 tahun tetapi juga mendukung perkembangan kognitif mereka secara keseluruhan.

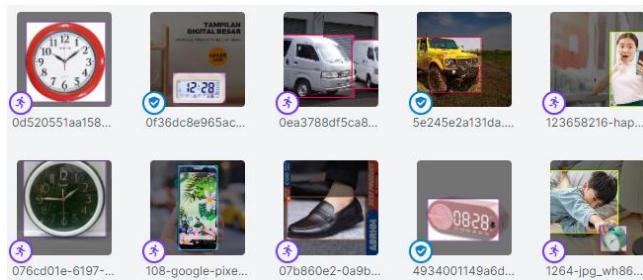
Sumber utama dari pengambilan gambar-gambar ini adalah internet. Gambar nantinya akan diambil secara *random* sesuai dengan kelas-kelasnya. Seperti yang terlihat pada **Gambar 3. 4**, data ini nantinya akan digabung menjadi

satu *dataset* yang akan diuji. Total data yang digunakan dalam penelitian ini adalah 1200 gambar di mana masing-masing kelas terdiri dari 200 gambar.

3.3. Pre-Processing

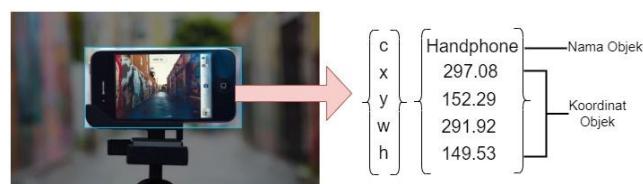
Dalam *pre-processing* gambar yang berguna untuk mengoptimalkan pelatihan model yang nantinya akan digunakan, terdapat beberapa proses.

3.3.1. Labeling Gambar



Gambar 3. 5: Contoh gambar yang telah dilakukan labeling

Labeling adalah tahap awal untuk menentukan objek dari gambar yang mau dilatih. Dalam proses ini, tiap gambar akan dibuat *bounding box* dengan menggunakan *roboflow* yang dapat diakses dari link <https://roboflow.com>. Seperti yang sudah dijelaskan pada poin mengenai YOLO, *bounding box* ini berguna untuk menentukan objek-objek dalam sebuah gambar. Setelah dilakukan *bounding box*, maka akan tersimpan data yang mencatat nilai-nilai yang menunjuk keterangan dan koordinat dari objek tersebut seperti yang terlihat pada **Gambar 3. 6**.



Gambar 3. 6: Ilustrasi nilai bounding box

3.3.2. Resize Image



Gambar 3. 7: Contoh hasil rescale

Pada tahap ini, gambar akan di-*resize* dengan ukuran 416 x 416. Tahap ini dilakukan agar gambar mempunyai ukuran yang seragam sebelum masuk ke dalam tahap *training* atau pelatihan. Penentuan pilihan ukuran citra ini didasari pada penelitian yang dilakukan oleh Liquan Zhao (Zhao & Li, 2020) yang mendapatkan performa yang bagus dalam mendekripsi objek. Selain itu, ukuran ini pun sering digunakan pada pelatihan YOLO di beberapa penelitian sebelumnya.

3.3.3. Augmentasi



Gambar 3. 8: Contoh augmentasi yang diterapkan pada citra

Dalam augmentasi gambar akan melewati beberapa proses. Proses-proses ini diambil berdasarkan keputusan dari uji coba dengan beberapa sampel gambar. Adapun proses-proses augmentasi tersebut adalah *grayscale*, *exposure* (15%), *rotation* (10%), dan *flip* (horizontal).

Proses augmentasi ini bertujuan untuk menambah variasi dan jumlah data. Sehingga dapat mengoptimalkan dalam proses *training* dan menghindari *overfitting* (Fadillah dkk., 2021; Perez dkk., 2018; Zhao & Li, 2020).

3.3.4. Data Split

Pada penelitian ini, *dataset* yang telah melalui proses *resize* dan augmentasi akan masuk ke tahap pembagian dataset. Pada pembagian ini akan dibagi menjadi tiga bagian, yaitu *training*, *validation* dan *testing* dengan pembagian 70:20:10. Dikarenakan menggunakan Roboflow, maka ketika dataset terbentuk, Roboflow akan secara otomatis membagi dataset menjadi tiga bagian. (Kumar dkk., 2021; Mohammed & Ekmekci, 2024).

3.4. Mengimport Dataset Ke Proyek

Dataset yang telah didapatkan dari internet dan diolah di Roboflow akan diimporkan ke dalam proyek. Untuk mengimportnya pun dapat dilakukan dengan dua cara, yaitu dengan menggunakan *code* yang nantinya akan dimasukkan ke dalam proyek yang dibuat, atau dengan cara mendownload langsung datasetnya (Dwyer & Gallagher, 2023).

3.5. Modeling

Dalam proses modeling akan dilakukan beberapa tahap seperti pada **Gambar 3. 3.**

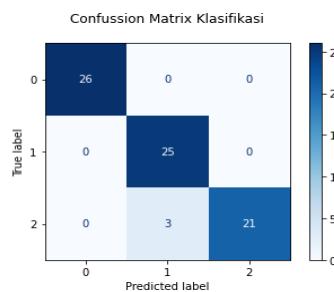
3.5.1. Mendapatkan Dependency Yolo Versi 8

Dependency merupakan hal yang diperlukan bagi program agar dapat dijalankan dengan baik. Untuk mendapatkan *dependency* YOLO versi 8, pada tahap ini akan dilakukan instalasi *library* yang telah disiapkan oleh pengembang. Selain itu, *dependency* YOLO juga bisa didapatkan dengan menduplikat *repository* yang sudah disiapkan pengembang dan menginstall persyaratannya (Jocher & Sergiuwaxmann, 2023).

3.5.2. Pelatihan Model

Pada tahap ini, dataset yang telah diimpor dari Roboflow akan dilatih dengan *dependency* dari YOLO versi 8 yang sebelumnya sudah didapatkan. Proses ini akan melibatkan beberapa parameter yang dapat mempengaruhi kinerja, kecepatan, dan akurasi model (Jocher & Sergiuwaxmann, 2023).

3.6. Evaluate



Gambar 3. 9: Contoh confussion matrix

Sumber: <https://haloryan.com/blog/cara-membaca-confusion-matrix>

Untuk mengetahui hasil pelatihan model pada tahap sebelumnya, maka pada tahap selanjutnya akan dilakukan evaluasi. Evaluasi dilakukan dengan melihat hasil *confusion matrix* yang berguna untuk memantau kesalahan kelas prediksi yang dilakukan model. Selain itu, evaluasi juga dilakukan dengan melihat grafik dari *mean average precision* (mAP) yang dapat menentukan sebaik apa model dapat mengenali objek.

3.7. Langkah – Langkah atau Alur Arsitektur YOLO

Pada poin ini akan membahas mengenai alur atau langkah-langkah dari arsitektur YOLO.

3.7.1. Backbone dan Neck

Pada bagian *backbone* dan *head* akan terjadi proses *feature extraction* yang bertujuan untuk mendapatkan fitur-fitur dari gambar inputan. Pada proses ini,

setelah gambar diinputkan maka gambar akan melibatkan beberapa langkah penting.

3.7.1.1. Convolutional Layer

Pada tahap ini, akan dilakukan proses ekstraksi fitur dari inputan yang didapatkan dari tahap sebelumnya. Berikut pada **Gambar 3. 15.** adalah contoh *matrix pixel* yang berukuran 5x5 dengan *kernel* 3x3.

Pada tahap awal dari *convolution* ini adalah melakukan perkalian antara *channel matrix* dengan *kernel*. Proses perkalian ini akan dilakukan berulang dengan

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

1	0	1
0	1	0
1	0	1

Kernel 3x3

Gambar 3. 15: Contoh matrix inputan 5x5 dan kernel 3x3

menggeserkan *kernel* sebanyak satu *strides* tiap *channel*-nya. Sehingga untuk hasil dari tahap ini dapat dilihat pada ilustrasi di bawah.

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output (1,1):
 $(15*1)+(0*0)+(0*1)+(15*0)+(12*1)+(15*0)+(15*1)+(5*0)+(12*1)$
 $= 54$

Gambar 3. 14: Output channel 1,1

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output (2,2):
 $(12*1)+(15*0)+(15*1)+(5*0)+(12*1)+(0*0)+(8*1)+(15*0)+(15*1)$
 $= 62$

Gambar 3. 10: Output channel 2,2

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output (2,3):
 $(12*1)+(15*0)+(15*1)+(5*0)+(12*1)+(0*0)+(8*1)+(15*0)+(15*1)$
60

Gambar 3. 20: Output channel 2,3

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output 3,1:
 $(15*1)+(5*0)+(12*1)+(15*0)+(8*1)+(15*0)+(15*1)+(10*0)+(13*1)$
63

Gambar 3. 19: Output channel 3,1

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output 3,2:
 $(5*1)+(12*0)+(0*1)+(8*0)+(15*1)+(15*0)+(10*1)+(13*0)+(0*1)$
30

Gambar 3. 18: Output channel 3,2

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15

Output 3,3:
 $(12*1)+(0*0)+(15*1)+(15*0)+(15*1)+(15*0)+(13*1)+(0*0)+(15*1)$
70

Gambar 3. 17: Output channel 3,3

Setelah dilakukan perhitungan *convolution* maka hasil yang didapatkan dapat dilihat dari **Gambar 3. 16.**

15	0	0	13	15
15	12	15	15	15
15	5	12	0	15
15	8	15	15	15
15	10	13	0	15



54	33	57
65	62	60
63	30	70

Gambar 3. 16: Hasil convolution

Setelah melewati proses konvolusi, maka berikutnya akan melewati *activation layers*. Misalkan *activation layers* yang digunakan adalah *activation* ReLU (*Rectified Linear Unit*), di mana $\text{ReLU}(x) = \max(0, x)$ maka hasil dari

proses ini akan sama dengan hasil dari konvolusi, karena ReLU akan mengganti semua nilai negatif dengan nol dan mempertahankan semua nilai positif tanpa perubahan (Putra dkk., 2016; Sari dkk., 2022).

3.7.1.2. Pooling Layer

Setelah melewati proses *convolution*, maka akan masuk pada proses selanjutnya, yaitu pooling layer. Untuk jenis *pooling layer* yang digunakan pada contoh ini merupakan jenis *max pooling* yang mengambil nilai maksimum dari tiap matriks (Dwi Antoko dkk., 2021). Karena hasil konvolusi sebelumnya adalah matriks 3x3, untuk mempermudah *max pooling* dengan *kernel* 2x2, maka akan dilakukan *padding* pada sisi atas dan kiri dengan nilai nol. *Padding* dilakukan untuk menambahkan baris dan kolom nol di bagian atas dan kiri, sehingga matriks menjadi 4x4 seperti pada **Gambar 3. 21**.

54	33	57	
65	62	60	
63	30	70	

→

0	0	0	0
0	54	33	57
0	65	62	60
0	63	30	70

Gambar 3. 21: Hasil padding dengan nilai 0

Selanjutnya akan dilakukan proses *max pooling* seperti ilustrasi di bawah ini.

0	0	0	0
0	54	33	57
0	65	62	60
0	63	30	70

→

54	
	57

0	0	0	0
0	54	33	57
0	65	62	60
0	63	30	70

→

54	57
65	70

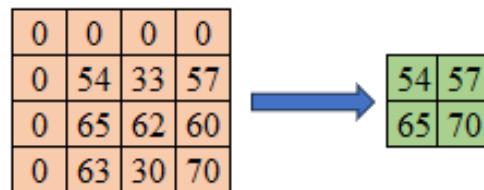
0	0	0	0
0	54	33	57
0	65	62	60
0	63	30	70

→

54	57
65	70

Gambar 3. 22: Proses dari max pooling

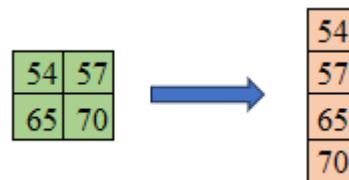
Hasil dari *max pooling* dapat dilihat pada **Gambar 3. 23**.



Gambar 3. 23: Hasil dari max pooling 2x2

3.7.1.3. Fully Connected Layer

Setelah melewati proses *pooling layer*, maka step selanjutnya akan masuk pada langkah *fully connected layer*. Pada tahap ini awalnya matriks dari *pooling layer* akan masuk pada proses *flatten* yang bertujuan untuk mengubah matriks dua dimensi menjadi matriks satu dimensi seperti yang terlihat pada **Gambar 3. 24**. Selanjutnya setelah matriks diubah, maka akan dilakukan perhitungan pada *dense* atau *hidden layer*. Untuk menghitung *dense layer*, di sini akan dibuat sebuah pemisalan untuk nilai bobot dan nilai bias, di mana:



Gambar 3. 24: Hasil flatten

$$w_1 = \begin{bmatrix} 0.2 & 0.4 & 0.6 \\ 0.8 & 0.2 & 0.4 \\ 0.5 & 0.7 & 0.3 \\ 0.9 & 0.1 & 0.2 \end{bmatrix}$$

$$b_1 = [0.1 \quad 0.1 \quad 0.1]$$

$$w_2 = \begin{bmatrix} 0.3 & 0.2 & 0.5 & 0.9 \\ 0.4 & 0.6 & 0.1 & 0.2 \\ 0.7 & 0.3 & 0.8 & 0.5 \end{bmatrix}$$

$$b_2 = [0.2 \quad 0.2 \quad 0.2]$$

Selanjutnya akan dilakukan perhitungan terhadap inputan *hidden layer* dengan menggunakan persamaan 4.

$$z_1 = x \cdot w_1 + b_1 \tag{4}$$

Sehingga menjadi:

$$z_1 = [54 \ 57 \ 65 \ 70] \cdot \begin{bmatrix} 0.2 & 0.4 & 0.6 \\ 0.8 & 0.2 & 0.4 \\ 0.5 & 0.7 & 0.3 \\ 0.9 & 0.1 & 0.2 \end{bmatrix} + [0.1 \ 0.1 \ 0.1]$$

$$z_1 = [54 \cdot 0.2 + 57 \cdot 0.8 + 65 \cdot 0.5 + 70 \cdot 0.9, 54 \cdot 0.4 + 57 \cdot 0.2 + 65 \cdot 0.3 + 70 \cdot 0.1, 54 \cdot 0.6 + 57 \cdot 0.4 + 65 \cdot 0.1 + 70 \cdot 0.2] + [0.1, 0.1, 0.1]$$

$$z_1 = [108 + 45.6 + 32.5 + 63 + 0.1, 21.6 + 11.4 + 45.5 + 7 + 0.1, 32.4 + 22.8 + 19.5 + 14 + 0.1]$$

$$z_1 = [249.2, 85.6, 88.8]$$

Setelah didapatkannya nilai z_1 maka selanjutnya akan diterapkan *activation ReLU*.

Namun karena nilai z_1 bernilai positif, maka hasil dari perhitungan ReLU pun akan tetap sama dengan nilai z_1 sebelumnya. Untuk itu selanjutnya z_1 akan menjadi nilai a_1 . Berikutnya akan dilakukan perhitungan terhadap *hidden layer* ke *output layer* dengan persamaan 5.

$$z_2 = a_1 \cdot w_2 + b_2 \quad (5)$$

Maka untuk hasilnya akan menjadi:

$$z_2 = [249.2 \ 85.6 \ 88.8] \cdot \begin{bmatrix} 0.3 & 0.2 & 0.5 & 0.9 \\ 0.4 & 0.6 & 0.1 & 0.2 \\ 0.7 & 0.3 & 0.8 & 0.5 \end{bmatrix} + [0.2 \ 0.2 \ 0.2]$$

$$z_2 = [249.2 \cdot 0.3 + 85.6 \cdot 0.4 + 88.8 \cdot 0.7, 249.2 \cdot 0.2 + 85.6 \cdot 0.6 + 88.8 \cdot 0.3, 249.2 \cdot 0.5 + 85.6 \cdot 0.1 + 88.8 \cdot 0.8, 249.2 \cdot 0.9 + 85.6 \cdot 0.2 + 88.8 \cdot 0.5] + [0.2, 0.2, 0.2, 0.2]$$

$$z_2 = [74.76 + 34.24 + 62.16 + 0.2, 49.84 + 51.36 + 26.64 + 0.2, 124.6 + 8.56 + 71.04 + 0.2, 224.28 + 17.12 + 44.4 + 0.2]$$

$$z_2 = [171.36, 128.04, 204.4, 285.92]$$

Selanjutnya akan dilakukan perhitungan *softmax* menggunakan persamaan

$$\text{softmax}(z_2) = \frac{e^{z_2}}{\sum e^{z_2}} \quad (6)$$

Perhitungan *softmax* ini berfungsi untuk mengubah nilai keluaran atau nilai *output* untuk menjadi probabilitas yang memiliki nilai dari rentang 0 – 1 (Hardi & Sundari,

2022). Karena itu dari hasil z_2 akan dilakukan perhitungan *softmax* sehingga didapatkan:

Objek1	0
Objek2	0
Objek3	0
Objek4	1

Gambar 3. 25: Hasil softmax

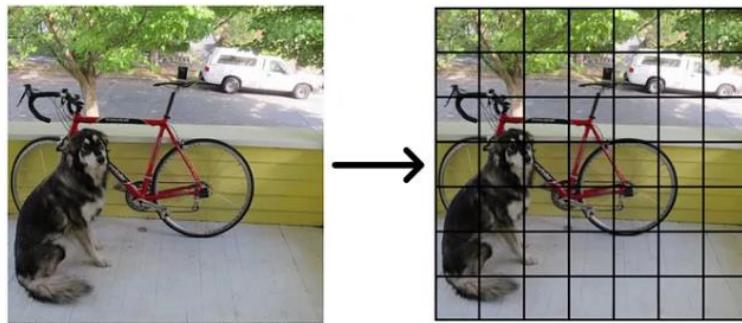
Dari **Gambar 3. 25** dapat dilihat bahwa inputan yang dimasukkan sebelumnya setelah melewati beberapa proses, ditemukan diklasifikasi atau dideteksi sebagai objek 4.

3.7.2. Head

Setelah inputan melewati *backbone* dan *neck* maka selanjutnya akan masuk pada *head* yang bertugas untuk mengubah hasil ekstraksi dan penyusunan layer menjadi prediksi *bounding box* dan kelas-kelas objek yang terdaftar di model. Ada beberapa proses yang terdapat pada *head*.

3.7.2.1. Pembagian Grid SxS

Model YOLO membagi gambar *input* menjadi *grid* berukuran $S \times S$. Setiap sel dalam *grid* ini akan bertanggung jawab untuk mendeteksi objek yang jatuh di dalam area sel tersebut. Pada *grid* ini, setiap sel akan memprediksi sejumlah *bounding box* dan skor kepercayaan (*confidence skor*) yang menunjukkan kemungkinan adanya objek dan jenis kelas objek tersebut (Redmon dkk., 2015; Terven & Cordova-Esparza, 2023).



Gambar 3. 26: Proses pembagian gambar menjadi grid SxS

Sumber: <https://medium.com/@ayushyajnik2/computer-vision-yolo-grid-cells-and-anchor-boxes-57b8a33cb25b>

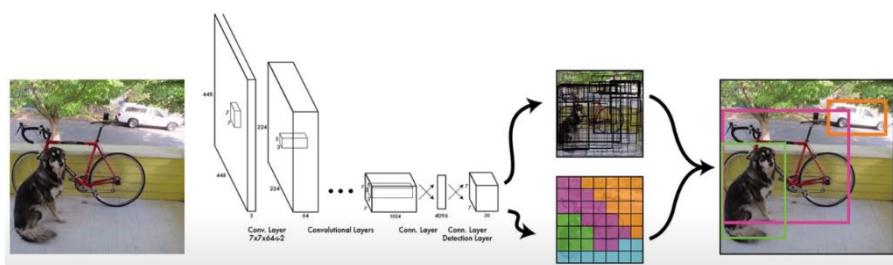
3.7.2.2. Thereshold dan Non-Maximum Suppression (NMS)

Setelah tiap *grid* memprediksi objek yang ada di dalamnya dan menghasilkan *bounding box* dan *confidence skor*, maka selanjutnya akan masuk pada proses *threshold*. *Thresholding* digunakan untuk menentukan apakah prediksi *bounding box* dianggap relevan atau tidak. Hanya *bounding box* dengan skor kepercayaan yang lebih tinggi dari nilai *threshold* yang akan dipertimbangkan lebih lanjut dalam proses prediksi. Proses ini akan membantu dalam mengurangi jumlah prediksi yang salah. Setelah proses *threshold*, maka akan masuk pada proses NMS.

Non-Maximum Suppression adalah teknik yang digunakan untuk mengurangi duplikasi prediksi *bounding box* yang mungkin terjadi di sekitar objek yang sama. NMS memilih *bounding box* dengan skor kepercayaan tertinggi dan menekan atau mengabaikan semua *bounding box* lainnya yang memiliki *overlap* (IOU - *Intersection Over Union*) di atas ambang batas tertentu dengan *bounding box* terpilih. Hal ini memastikan bahwa setiap objek terdeteksi hanya dengan satu *bounding box* yang paling akurat (Redmon dkk., 2015; Terven & Cordova-Esparza, 2023).

3.7.2.3. Output

Bagian terakhir dari *head* adalah menghasilkan *output* akhir berupa *bounding box* dengan skor kepercayaan dan kelas objek yang telah diolah melalui proses *thresholding* dan NMS. *Output* ini kemudian dapat digunakan untuk berbagai aplikasi seperti deteksi objek, pengenalan gambar, dan sebagainya (Redmon dkk., 2015; Terven & Cordova-Esparza, 2023). Secara ringkas untuk alur YOLO dapat dilihat pada **Gambar 3. 27**



Gambar 3. 27: Ilustrasi proses deteksi YOLO

Sumber: <https://www.youtube.com/watch?v=kMDf35Ta-84&list=WL&index=1>

3.8. Skenario Pengujian

Dalam penelitian ini akan dilakukan beberapa eksperimen dengan melibatkan parameter-parameter yang digunakan di dalam YOLO. Adapun parameter-parameter tersebut adalah *dropout*, *batch*, *learning rate*, dan *optimizer*. Sedangkan untuk datasetnya sendiri akan menggunakan dataset dengan ukuran gambar 416x416 yang memiliki augmentasi. Penggunaan parameter-parameter ini dilandasi dari beberapa penelitian yang sebelumnya sudah dilakukan dan mendapatkan hasil yang bagus (Gibran dkk., t.t.; Karna dkk., 2023; Li dkk., 2016; Sandhya & Kashyap, 2024; Sholahuddin dkk., 2023).

Secara lebih rinci terkait parameter dan nilai yang digunakan dapat dilihat pada **Tabel 3. 1**.

Tabel 3. 1: Tabel Skenario Pengujian

Parameter	Nilai Parameter	Hipotesis	Penjelasan
Model <i>Pre-Trained</i>	Yolov8s.pt	Penggunaan jenis model <i>pre-trained</i> berikut dapat membantu percepatan waktu komputasi (Jocher & Sergiuwaxmann, 2023).	Dataset telah melewati proses <i>resize</i> menjadi ukuran 416x416 dan ukuran tersebut masuk dalam kategori kecil hingga menengah (Banovbi dkk., 2022).
	Yolov8m.pt		
<i>Dropout</i>	0.2	Semakin besar nilai <i>dropout</i> , maka semakin banyak neuron yang dihilangkan secara acak pada setiap layer selama <i>training</i> (Wei dkk., 2020).	Hal ini dapat membantu mencegah <i>overfitting</i> dengan mengurangi interkoneksi antar neuron (Wei dkk., 2020).
	0.5		
<i>Batch</i>	32	Semakin besar nilai <i>batch</i> , maka semakin banyak data yang diproses secara bersamaan pada setiap <i>update</i> parameter model (Singh dkk., 2020).	Hal ini dapat meningkatkan efisiensi <i>training</i> dengan mengurangi <i>overhead</i> komputasi (Singh dkk., 2020).
	64		
<i>Learning rate</i>	0.001	Semakin tinggi nilai <i>learning rate</i> , maka semakin cepat model belajar dari data <i>training</i> (Wen dkk., 2021).	Hal ini dapat membantu model mencapai konvergensi dengan lebih cepat (Wen dkk., 2021).
	0.0001		
<i>Optimizer</i>	Adam	Adam dan RMSProp adalah dua <i>optimizer</i> yang berbeda dengan kelebihan dan kekurangannya masing-masing (Li dkk., 2016).	Adam umumnya dianggap lebih stabil dan konvergen lebih cepat, sedangkan RMSProp lebih baik dalam menangani data dengan gradien yang bervariasi (Li dkk., 2016).
	RMSProp		

3.9. Pengujian Kelayakan Kuesioner dan Kegunaan Hasil Akhir

3.9.1. Target Kuesioner

Berdasarkan batasan masalah yang telah dibuat, maka target dari kuesioner yang dibuat ini adalah orang tua yang memiliki anak usia 2 – 4, guru PAUD, dan beberapa masyarakat umum yang menjadi titik fokus pada penelitian ini.

3.8.2. Waktu dan Tempat Pengujian

Pengujian kelayakan hasil akhir ini akan dilakukan melalui kuesioner. Kuesioner ini nantinya akan dibagi menjadi dua bagian. Pertama, kuesioner dibuka untuk dilakukan pengujian validitas dan pengujian reliabilitas terhadap instrumen pengujian. Dan yang kedua, kuesioner akan disebarluaskan untuk diuji *usability* atau kegunaan dari aplikasi yang dihasilkan. Kuesioner akan dilakukan dengan menyebarkan *link* hasil akhir dan *link* kuesioner ke masyarakat umum secara *online* selama seminggu.

3.8.3. Uji Kelayakan Kuesioner

Uji kelayakan pada hasil penelitian ini berdasarkan aspek kegunaan dari penerapan algoritma You Only Look Once (YOLO) dalam pengenalan lingkungan sekitar bagi anak-anak. Tanggapan dari kuesioner ini akan mewakili empat sub aspek, yaitu pemahaman (*understandability*), pengoperasian (*operability*), daya tarik (*attractiveness*), dan kemampuan belajar (*learnability*). Adapun rincian dari keempat sub aspek tersebut dapat dilihat dari Error! Reference source not found. (Budiyanta, 2018; Ernawati & Sukardiyono, 2017).

Tabel 3. 2: Rincian pernyataan kuesioner

No.	Pernyataan	Sub Aspek
1	Sistem kerja website mudah untuk dipahami	<i>Understandability</i>
2	Navigasi atau arahan website ini mudah dipahami	<i>Understandability</i>
3	Website menyediakan informasi yang cukup jelas	<i>Understandability</i>
4	Website berfungsi secara efisien dalam pengenalan objek dilingkungan sekitar	<i>Operability</i>
5	Kesulitan dalam menemukan fitur-fitur yang dibutuhkan	<i>Operability</i>
6	Desain antarmuka dapat dipahami dengan mudah	<i>Attractiveness</i>
7	Penggunaan warna, gambar, dan elemen desain menarik perhatian	<i>Attractiveness</i>
8	Penilaian terhadap desain keseluruhan dari website	<i>Attractiveness</i>
9	Website sangat mudah untuk dipelajari	<i>Learnability</i>
10	Tata cara penggunaan dapat dengan mudah diingat	<i>Learnability</i>
11	Hasil deteksi sesuai dengan objek yang dideteksi	<i>Learnability</i>

Adapun ketentuan yang digunakan adalah ketentuan dari skala likert, sehingga ketentuan penilaian dapat dilihat pada **Tabel 3. 3.**

Tabel 3. 3: Ketentuan skala likert

Skala	Nilai
1	Sangat Tidak Setuju
2	Tidak Setuju
3	Setuju
4	Sangat Setuju

3.9.3.1. Uji Validitas

Uji validitas akan digunakan untuk mengetahui kelayakan dari butir-butir dalam pertanyaan yang mendefinisikan variabel. Metode uji validitas yang akan digunakan pada penelitian ini adalah metode korelasi produk momen (*corellate bivariate pearson*) dengan 5% untuk r tabel signifikan (Rosita dkk., 2021).

3.9.3.2. Uji Reliabilitas

Berdasarkan poin 2.2.13, maka suatu instrumen dianggap reliabel apabila instrumen tersebut dapat dipercaya sebagai alat ukur dari penelitian. Uji reliabilitas pada penelitian ini akan menggunakan metode Cronbach's Alpha dengan rumus yang sesuai pada persamaan 3. Untuk mengetahui tinggi rendahnya reliabilitas dapat menggunakan kategori seperti pada **Tabel 3. 4.**

Tabel 3. 4: Tingkat reliabilitas Cronbach's Alpha

Interval Reliabilitas	Kategori
$0,80 < r \leq 1,00$	Reliabilitas sangat tinggi
$0,60 < r \leq 0,80$	Reliabilitas tinggi
$0,40 < r \leq 0,60$	Reliabilitas sedang
$0,20 < r \leq 0,40$	Reliabilitas rendah

$0,00 < r \leq 0,20$	Tidak reliabel
----------------------	----------------

(Ernawati & Sukardiyono, 2017).

3.9.4. Analisis Kegunaan (*Usability*)

Setelah mengetahui bahwa instrumen yang hendak diukur adalah valid dan reliabilitas, maka akan dilakukan pengukuran kegunaan atau *usability*. Pengukuran *usability* akan dilakukan dengan menghitung persentasi jawaban dari responden dengan menggunakan rumus pada persamaan 4. Setelah data diperoleh, maka data akan dikonversi ke dalam kategorinya berdasarkan pada **Tabel 3. 5.**

Tabel 3. 5: Kategori Kegunaan

Data (%)	Kategori
< 20	Sangat tidak layak
21 – 40	Tidak layak
41 – 60	Cukup
61 – 80	Layak
81 – 100	Sangat Layak

(Kusuma dkk., 2016).

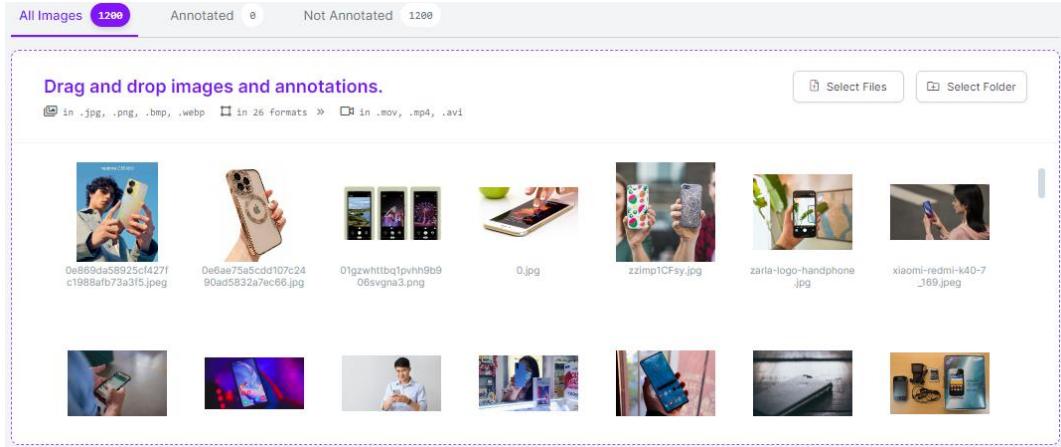
BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1. Pengumpulan Data

Data yang dikumpulkan adalah data yang diambil dari internet berupa gambar dan dipilih secara *random*. Data yang dikumpulkan ini kemudian dikelompokkan sesuai kelas-kelas yang nantinya akan digunakan pada penelitian seperti pada **Gambar 3.** 4. Setelah terkumpul, data akan dimasukkan ke dalam *workshop* yang sebelumnya sudah dibuat di Roboflow seperti yang dilihat pada

Gambar 4. 1. Dengan demikian, data akan siap masuk pada tahap *pre-processing*.

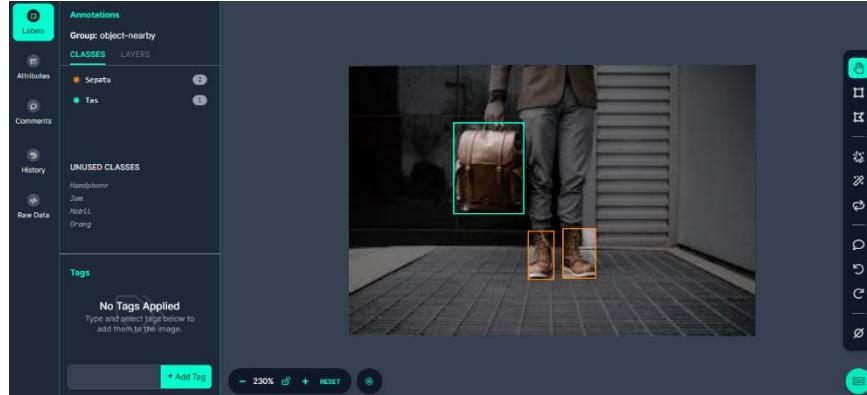


Gambar 4. 1: Proses upload data ke Roboflow

4.2. Pre-processing Data

Tahap *pre-processing* pada data akan melalui beberapa tahap seperti yang telah disebutkan pada bab 3. Pengolahan dataset ini akan dikerjakan menggunakan *tools* Roboflow mulai dari pelabelan gambar, *resize* gambar, augmentasi gambar, dan pembagian data.

4.2.1. Labeling Gambar



Gambar 4. 2: Proses annotate di Roboflow

Pada tahap ini, gambar yang sebelumnya sudah diunggah ke *workshop* Roboflow akan dilakukan *annotate* atau pelabelan objek yang terdapat di dalam gambar. Pelabelan pada sebuah gambar tidak terpaku hanya pada satu objek. Jika di dalam gambar terdapat lebih dari satu objek yang jelas, maka objek tersebut akan dilabeli. Hal ini dapat terlihat seperti pada **Gambar 4. 2**.

```
{
  "boxes": [
    {
      "label": "Tas",
      "x": "94.57",
      "y": "69.78",
      "width": "47.39",
      "height": "61.38"
    },
    {
      "label": "Sepatu",
      "x": "130.00",
      "y": "128.48",
      "width": "17.39",
      "height": "32.61"
    },
    {
      "label": "Sepatu",
      "x": "155.87",
      "y": "127.17",
      "width": "22.17",
      "height": "33.48"
    }
  ]
}
```

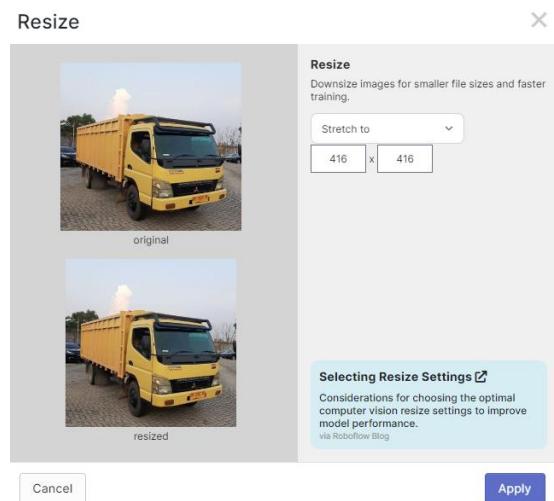
Gambar 4. 3: Data hasil labeling atau bounding box

Seperti yang dijelaskan pada 3.3.1 mengenai *labeling*, label atau *bounding box* yang dibuat ini akan menyimpan informasi-informasi yang dibutuhkan algoritma untuk proses pemodelan. Adapun data yang tersimpan seperti yang

terlihat pada **Gambar 4. 3**. Data tersebut memuat mengenai *class* dari objek yang dilabel dan koordinat dari objek tersebut.

4.2.2. Resize Image

Setelah dilabel, gambar akan masuk pada tahap selanjutnya, yaitu *resize*. *Resize* ini membuat semua gambar pada versi yang dipilih akan memiliki ukuran yang sama. Hal ini dilakukan agar dapat membantu meningkatkan efisiensi dalam pengolahan data dan penggunaan sumber daya komputasi pada saat pemodelan nantinya (Iryanto & Zaini, 2014; Zhao & Li, 2020).



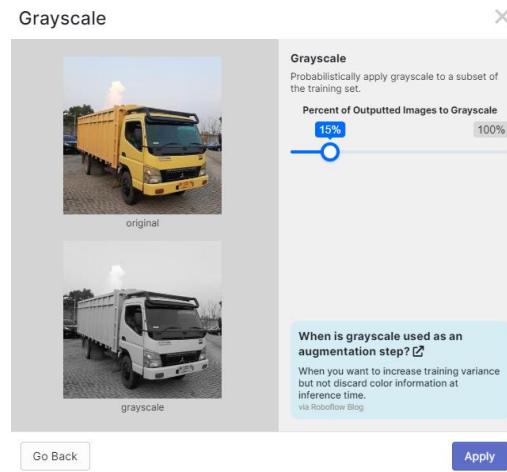
Gambar 4. 4: Proses resize pada dataset

4.2.3. Augmentasi Image

Pada tahap ini akan dilakukan proses augmentasi seperti yang sudah dijelaskan pada bab 3.

4.2.3.1. Grayscale

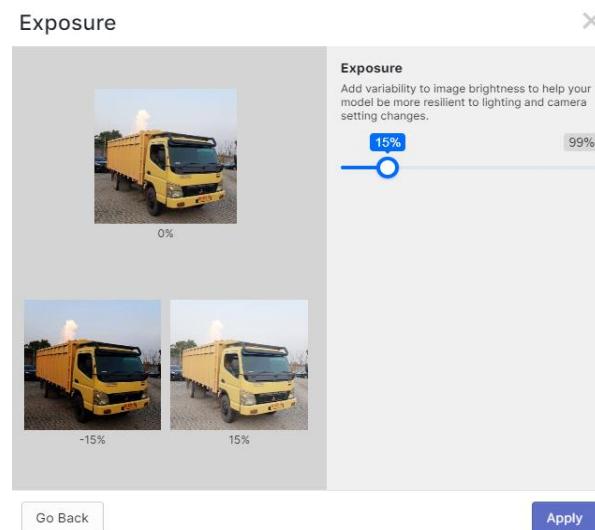
Pada augmentasi untuk *grayscale*, akan dipilih tingkat *grayscale* sebesar 15%. Tingkatan ini dapat membuat variasi dataset yang signifikan jika dibandingkan dengan tingkatan di bawahnya juga tidak membuat gambar terlalu gelap sehingga dapat menghilangkan detail dari gambar.



Gambar 4. 5: Proses augmentasi dataset

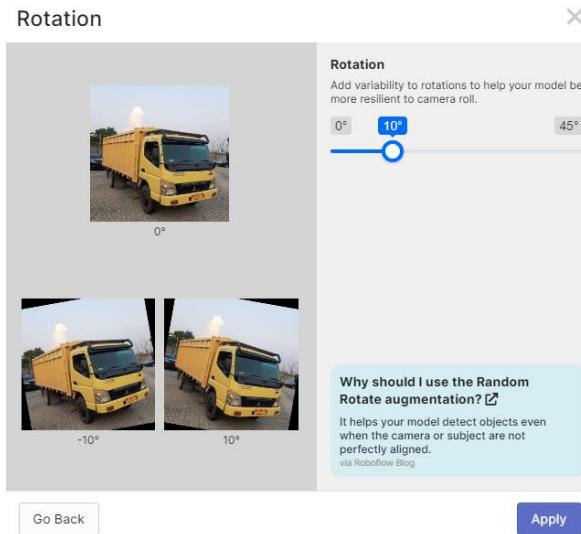
4.2.3.2. Exposure

Untuk bagian *exposure* dataset akan menggunakan tingkatan sebesar 15% sehingga membuat dataset menjadi lebih gelap dan lebih terang dengan nilai 15%.



Gambar 4. 6: Proses exposure dataset

4.2.3.3. Rotation

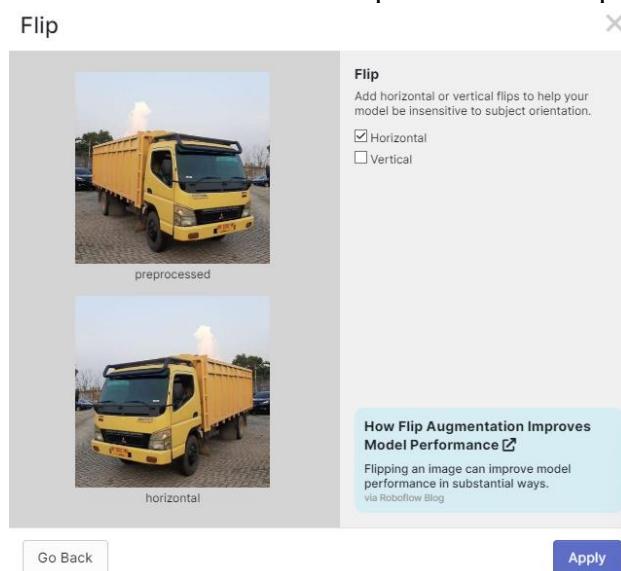


Gambar 4. 7: Proses rotation dataset

Pada tahap ini dataset akan dilakukan proses *rotation* dengan tingkatan sebesar 10%. Hal ini dilakukan agar terdapat beberapa variasi dataset yang memiliki titik yang berbeda dari gambar aslinya. *Rotation* ini akan membuat gambar memutar dengan sejumlah derajat tertentu (Perez dkk., 2018).

4.2.3.4. Flip

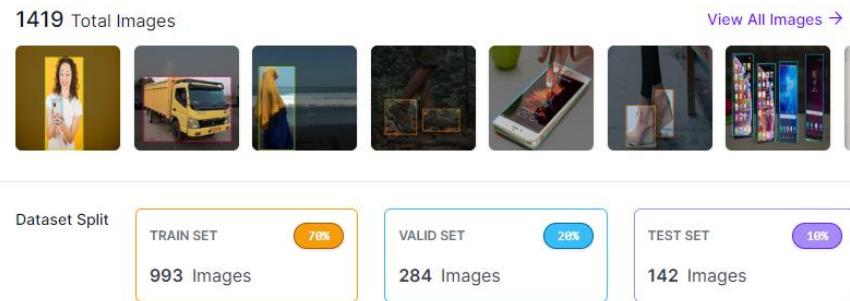
Proses ini nantinya akan membalikan gambar secara horizontal. Proses ini digunakan agar dataset memiliki variasi terhadap data dari beberapa sudut pandang.



Gambar 4. 8: Proses flip dataset

4.2.4. Data Split

Pada tahap *data split*, setelah tahapan *resize* dan augmentasi dilakukan, maka secara otomatis Roboflow akan menghitung total gambar kemudian membaginya ke dalam bagianya masing-masing. Untuk pembagiannya akan menerapkan perbandingan 70:20:10 dengan rincian 70% untuk *train*, 20% untuk *valid*, dan 10% untuk *testing* seperti yang dijelaskan pada bagian sebelumnya.



Gambar 4. 9: Proses pembagian dataset

Ditahap ini dataset telah selesai diolah dan siap digunakan ke dalam proyek yang akan dibuat untuk digunakan dalam pelatihan model.

4.3. Impor Dataset

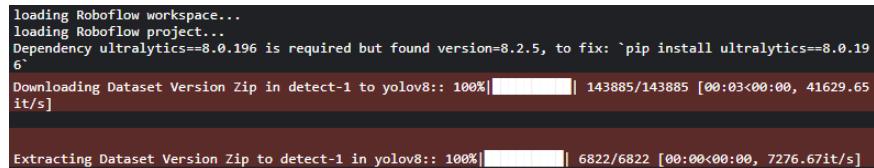
```
!pip install roboflow
from roboflow import Roboflow

rf = Roboflow(api_key="A88aglIXim3OCSF5vNOe")
project = rf.workspace("gabriel-advent-hobto").project(
    "detect-tdz36")
version = project.version(1)
dataset = version.download("yolov8")
```

Gambar 4. 10: Code untuk mengimpor dataset dari Roboflow

Dataset yang sebelumnya sudah diolah akan diimpor ke dalam proyek untuk membantu pelatihan model dengan *custom* dataset. Untuk mengimpor dataset, Roboflow sendiri telah memberikan kemudahan dengan *code* yang sudah disiapkan seperti yang dapat dilihat dari **Gambar 4. 10**. *Code* yang diberikan ini nantinya akan mengunduh dalam bentuk zip dan mengekstrak dataset tersebut agar bisa

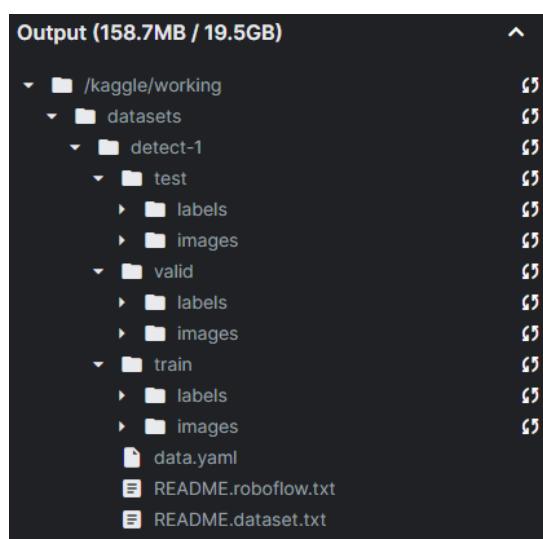
dipakai. Jika dataset yang diimpor berhasil diunduh dan diekstrak dari bentuk zip-nya maka pada *output* yang dihasilkan akan terlihat seperti pada **Gambar 4. 11**.



```
loading Roboflow workspace...
loading Roboflow project...
Dependency ultralytics==8.0.196 is required but found version=8.2.5, to fix: `pip install ultralytics==8.0.196`
Downloading Dataset Version Zip in detect-1 to yolov8::: 100%|██████████| 143885/143885 [00:03<00:00, 41629.65 it/s]
Extracting Dataset Version Zip to detect-1 in yolov8::: 100%|██████████| 6822/6822 [00:00<00:00, 7276.67it/s]
```

Gambar 4. 11: Keterangan jika berhasil mengunduh dan mengekstrak dataset

Setelah berhasil diekstrak, dapat dilihat pada **Gambar 4. 12** bahwa Roboflow telah membagikan dataset ke dalam foldernya masing-masing sesuai pembagian yang ada di bagian 4.2.4 sebelumnya mengenai *data split*. Pembagian dataset ini juga telah dilengkapi dengan label yang memiliki *extention .txt*. File inilah yang berisi data-data objek dari gambar sesuai pelabelan yang telah dilakukan. Selain itu, Roboflow juga telah membuat *file* data.yaml. *File* ini digunakan untuk mengonfigurasi dataset. Sehingga *file* data.yaml ini akan berisi informasi mengenai *path* dataset, kelas-kelas yang digunakan dalam model nantinya, dan beberapa informasi yang relevan. Secara lebih jelas terkait isi dari *file* data.yaml, dapat dilihat dari **Gambar 4. 13**.



Gambar 4. 12: Hasil ekstrasi dataset

```
train: ../train/images
val: ../valid/images
test: ../test/images

nc: 6
names: ['Handphone', 'Jam', 'Mobil', 'Orang', 'Sepatu', 'Tas']

roboflow:
  workspace: gabriel-advent-hobto
  project: detect-tdz36
  version: 1
  license: CC BY 4.0
  url: https://universe.roboflow.com/gabriel-advent-hobto/detect-tdz36/dataset/1
```

Gambar 4. 13: Isi dari file data.yaml

4.4. Modeling

Proses pembuatan model dari YOLO ini nantinya akan dibuat dengan bantuan Kaggle Notebook. Hal ini dikarenakan platform ini menyediakan akses GPU dengan *limit* 30 jam setiap minggunya. Meskipun ada batasan, fitur ini sangat menguntungkan karena mampu menjalankan program yang membutuhkan GPU di dalamnya (A. Y. Wang dkk., 2021).

4.4.1. Mendapatkan Depedency YOLO

```
pip install ultralytics
```

Gambar 4. 14: Code untuk menginstall library

Proses pembuatan model yang pertama kali dilakukan adalah dengan mendapatkan *depedency* YOLO terbaru yang sudah disiapkan oleh Ultralytics. Untuk mendapatkannya dapat dengan menginstall *library* yang sudah disiapkan seperti pada **Gambar 4. 14**. Untuk memastikan apakah *library* telah di *install* dapat dengan menjalankan *code* pada **Gambar 4. 15** yang nantinya akan menampilkan

```
import ultralytics
ultralytics.checks()
```

Gambar 4. 15: Code untuk mengecek ultralytics

output versi dari *library*, python, dan pytorch seperti pada **Gambar 4. 16**. Dengan *output* seperti pada gambar tersebut dapat dipastikan bahwa proyek yang dibuat

telah memiliki *dependency* dari beberapa *library* yang menjadi *requirements* untuk menjalannya YOLO seperti PyTorch, torchvision, OpenCV, numpy, pandas, tqdm, matplotlib, dan scipy.

```
Ultralytics YOLOv8.0.20 🚀 Python-3.10.13 torch-2.1.2 CUDA:0 (Tesla T4,  
15102MiB)  
Setup complete ✅ (4 CPUs, 31.4 GB RAM, 5689.3/8062.4 GB disk)
```

Gambar 4. 16: Output dari code untuk mengecek ultralytics

Tahap selanjutnya yang akan dilakukan agar dapat menggunakan YOLO adalah dengan mengimpor YOLO ke dalam proyek yang dibuat dengan menjalankan *code* seperti pada **Gambar 4. 17**. Dengan menjalankan kedua *code* tersebut, YOLO telah siap digunakan ke dalam proyek untuk pembuatan model. Ketika telah menjalankan *code* pada **Gambar 4. 17**, maka proses untuk mendapatkan *dependency* dari YOLO telah selesai dan siap untuk masuk pada tahap pelatihan model.

```
from ultralytics import YOLO
```

Gambar 4. 17: Mengimpor YOLO ke dalam projek

4.4.2. Pelatihan Model

Pada pelatihan model ini akan dilakukan beberapa percobaan berdasarkan tabel skenario pengujian. Adapun total keseluruhan percobaan ini adalah 32 percobaan. 32 percobaan ini merupakan hasil kombinasi yang dilakukan seperti pada *code* di **Gambar 4. 18**. Adapun rincian kombinasi tersebut dapat dilihat pada

Tabel 4. 1.

```
import subprocess  
from itertools import product  
  
param_grid = {  
    'model': ['yolov8s.pt', 'yolov8m.pt'],  
    'dropout': [0.2, 0.5],  
    'batch': [32, 64],  
    'learning_rate': [0.001, 0.0001],  
    'optimizer' : ['adam', 'RMSProp'],  
}  
  
# Generate semua kombinasi  
param_combinations = list(product(*param_grid.values()))
```

Gambar 4. 18: Code untuk mengombinasikan parameter

Tabel 4. 1: Daftar kombinasi parameter

No	Model	Dropout	Batch	Learning Rate	Optimizer
1	yolov8m	0,2	32	0,001	Adam
2	yolov8m	0,2	32	0,001	RMSProp
3	yolov8m	0,2	32	0,0001	Adam
4	yolov8m	0,2	32	0,0001	RMSProp
5	yolov8m	0,2	64	0,001	Adam
6	yolov8m	0,2	64	0,001	RMSProp
7	yolov8m	0,2	64	0,0001	Adam
8	yolov8m	0,2	64	0,0001	RMSProp
9	yolov8m	0,5	64	0,001	Adam
10	yolov8m	0,5	32	0,001	RMSProp
11	yolov8m	0,5	32	0,0001	Adam
12	yolov8m	0,5	32	0,0001	RMSProp
13	yolov8m	0,5	64	0,001	Adam
14	yolov8m	0,5	64	0,001	RMSProp
15	yolov8m	0,5	32	0,0001	Adam
16	yolov8m	0,5	64	0,0001	RMSProp
17	yolov8s	0,2	32	0,001	Adam
18	yolov8s	0,2	32	0,001	RMSProp
19	yolov8s	0,2	32	0,0001	Adam
20	yolov8s	0,2	32	0,0001	RMSProp
21	yolov8s	0,2	64	0,001	Adam
22	yolov8s	0,2	64	0,001	RMSProp
23	yolov8s	0,2	64	0,0001	Adam
24	yolov8s	0,2	64	0,0001	RMSProp
25	yolov8s	0,5	64	0,001	Adam
26	yolov8s	0,5	32	0,001	RMSProp
27	yolov8s	0,5	32	0,0001	Adam
28	yolov8s	0,5	32	0,0001	RMSProp
29	yolov8s	0,5	64	0,001	Adam
30	yolov8s	0,5	64	0,001	RMSProp
31	yolov8s	0,5	32	0,0001	Adam
32	yolov8s	0,5	64	0,0001	RMSProp

Pada pelatihan ini akan menggunakan 200 *epoch* dan menerapkan mekanisme *early stopping* dengan mengatur parameter *patience* ke 50. Hal ini memungkinkan pelatihan berhenti lebih awal untuk menghemat waktu komputasi jika model tidak menunjukkan perbaikan dalam jangka waktu tersebut. Sebelum dilakukan pelatihan, hasil kombinasi dari *code* di **Gambar 4. 18** akan dimasukkan ke dalam perintah-perintah untuk pelatihan dengan YOLO menggunakan *code* pada **Gambar 4. 19**.

```
commands = []
for params in param_combinations:
    model, dropout, batch, lr, optimizer = params
    command = f"yolo detect train \
data='/kaggle/working/datasets/detect-2/data.yaml' \
model='{model}' \
epochs=200 \
imgsz='416' \
optimizer='{optimizer}' \
lr0='{lr}' \
batch='{batch}' \
dropout='{dropout}' \
plots=True \
save=True"
    commands.append(command)
```

Gambar 4. 19: Code untuk melakukan pelatihan model pada YOLO

Pada *code* di **Gambar 4. 19** nantinya akan menyimpan susunan *code* seperti pada gambar 4.20 dengan kombinasi hasil dari **Gambar 4. 18** ke dalam variabel *commands*. Setelah proses pembentukan *code* seperti pada **Gambar 4. 20** selesai dilakukan, maka akan masuk pada tahap berikutnya, yaitu menjalankan perintah-perintah yang ada di variabel *commands* tersebut dengan menggunakan *code* pada **Gambar 4. 22**.

```
!yolo detect train \
data='/kaggle/working/datasets/detect-2/data.yaml' \
model=yolov8m.pt \
epochs=200 \
imgsz=416 \
optimizer='Adam' \
batch=64 \
lr0=0.0001 \
dropout=0.2 \
plots=True \
save=True
```

Gambar 4. 20: Contoh *code* yang disimpan di variabel array *commands*

Ketika *code* pada **Gambar 4. 22** dijalankan, maka program akan menampilkan kombinasi yang akan dijalankan seperti pada **Gambar 4. 21**.

```
for command in commands:
    print(command)
    subprocess.run(command, shell=True)
```

Gambar 4. 22: Code untuk menjalankan pelatihan sesuai dengan isi dari variabel commands

```
/kaggle/working
yolo detect train data='/kaggle/working/datasets/detect-2/data.yaml' model='yolov8s.pt' epochs=200 imgsz='416' optimizer=Adam lr0=0.001 batch='32' dropout='0.2' plots=True save=True
```

Gambar 4. 21: Output dari perintah 'print(command)'

Selama pelatihan, perkembangan model dapat diperhatikan dengan melihat proses pelatihan di tiap *epoch* seperti pada **Gambar 4. 23**. Berdasarkan **Gambar 4. 23**, dapat dilihat khususnya pada *epoch* 1/200, telah disajikan beberapa data seperti *training losses* yang dapat dilihat dari nilai *box_loss*, *cls_loss*, dan *dfl_loss*, yang secara berurutan akan melihat kehilangan (*loss*) dari prediksi *bounding box*, klasifikasi objek, dan distribusi prediksi. Selain itu, terdapat juga data mengenai kinerja model yang ditunjukkan dengan nilai *precision* (P), *recall* (R), mAP50 (50 berarti menggunakan nilai IoU sebesar 50%), dan mAP50-95 (50-95 berarti menggunakan nilai IoU sebesar 50-95%).

Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
1/200	3.96G	1.357	1.912	1.404	17	416: 100% [██████] 94/94 [00:30<00:00, 3.03it/s]
	Class	Images	Instances	Box(P R		mAP50 mAP50-95): 100% [██████] 5/5 [00:04<00:00, 1.17it/s]
	all	284	584	0.621	0.504	0.497 0.253
	Handphone	284	150	0.817	0.667	0.729 0.433
	Jam	284	40	0.525	0.7	0.613 0.309
	Mobil	284	75	0.624	0.56	0.556 0.289
	Orang	284	124	0.635	0.337	0.43 0.175
	Sepatu	284	134	0.254	0.53	0.262 0.0874
	Tas	284	61	0.874	0.23	0.394 0.221
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
2/200	5.68G	1.369	1.482	1.436	24	416: 100% [██████] 94/94 [00:27<00:00, 3.40it/s]
	Class	Images	Instances	Box(P R		mAP50 mAP50-95): 100% [██████] 5/5 [00:02<00:00, 1.71it/s]
	all	284	584	0.61	0.597	0.607 0.344
	Handphone	284	150	0.747	0.609	0.729 0.454
	Jam	284	40	0.707	0.6	0.726 0.434
	Mobil	284	75	0.788	0.742	0.769 0.483
	Orang	284	124	0.431	0.605	0.532 0.204
	Sepatu	284	134	0.545	0.403	0.44 0.248
	Tas	284	61	0.44	0.623	0.445 0.243
Epoch	GPU_mem	box_loss	cls_loss	dfl_loss	Instances	Size
3/200	5.68G	1.365	1.437	1.439	7	416: 100% [██████] 94/94 [00:27<00:00, 3.39it/s]
	Class	Images	Instances	Box(P R		mAP50 mAP50-95): 100% [██████] 5/5 [00:02<00:00, 1.77it/s]
	all	284	584	0.589	0.484	0.495 0.256
	Handphone	284	150	0.767	0.553	0.654 0.403
	Jam	284	40	0.391	0.3	0.382 0.225
	Mobil	284	75	0.788	0.596	0.62 0.332

Gambar 4. 23: Output pelatihan tiap epoch

4.5. Analisis Hasil Pengujian

Pelatihan dengan 32 kombinasi telah selesai dilakukan dan telah dirangkum dengan melihat nilai mAP pada **Tabel 4. 2.**

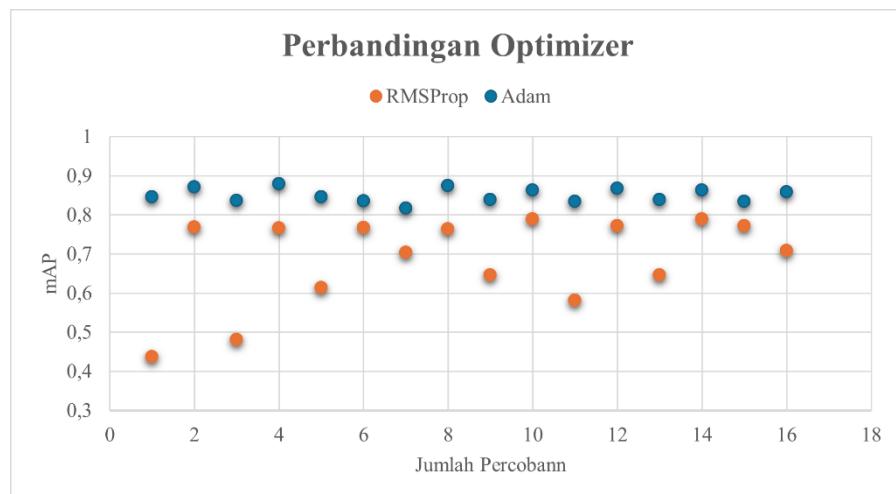
Tabel 4. 2: Hasil pelatihan model

No	Model	Dropout	Batch	Learning Rate	Optimizer	mAP
1	yolov8m	0,2	32	0,001	Adam	0,84499
2	yolov8m	0,2	32	0,001	RMSProp	0,43691
3	yolov8m	0,2	32	0,0001	Adam	0,87112
4	yolov8m	0,2	32	0,0001	RMSProp	0,7682
5	yolov8m	0,2	64	0,001	Adam	0,83652
6	yolov8m	0,2	64	0,001	RMSProp	0,48063
7	yolov8m	0,2	64	0,0001	Adam	0,87916
8	yolov8m	0,2	64	0,0001	RMSProp	0,76636
9	yolov8m	0,5	64	0,001	Adam	0,84499
10	yolov8m	0,5	32	0,001	RMSProp	0,6132
11	yolov8m	0,5	32	0,0001	Adam	0,83462
12	yolov8m	0,5	32	0,0001	RMSProp	0,7672
13	yolov8m	0,5	64	0,001	Adam	0,81623
14	yolov8m	0,5	64	0,001	RMSProp	0,70356
15	yolov8m	0,5	32	0,0001	Adam	0,87481
16	yolov8m	0,5	64	0,0001	RMSProp	0,7635
17	yolov8s	0,2	32	0,001	Adam	0,8386
18	yolov8s	0,2	32	0,001	RMSProp	0,64615
19	yolov8s	0,2	32	0,0001	Adam	0,86293
20	yolov8s	0,2	32	0,0001	RMSProp	0,78881
21	yolov8s	0,2	64	0,001	Adam	0,83396
22	yolov8s	0,2	64	0,001	RMSProp	0,58158
23	yolov8s	0,2	64	0,0001	Adam	0,86755
24	yolov8s	0,2	64	0,0001	RMSProp	0,77183
25	yolov8s	0,5	64	0,001	Adam	0,8386
26	yolov8s	0,5	32	0,001	RMSProp	0,64615
27	yolov8s	0,5	32	0,0001	Adam	0,86293
28	yolov8s	0,5	32	0,0001	RMSProp	0,78881
29	yolov8s	0,5	64	0,001	Adam	0,83396
30	yolov8s	0,5	64	0,001	RMSProp	0,77103
31	yolov8s	0,5	32	0,0001	Adam	0,85842
32	yolov8s	0,5	64	0,0001	RMSProp	0,70814

Dari hasil ke-32 percobaan pada **Tabel 4. 2** tersebut, diketahui bahwa percobaan yang mendapatkan hasil tertinggi terdapat pada percobaan ketujuh yang menghasilkan mAP sebesar 0,879 atau sekitar 87% dan dengan kombinasi parameter model jadi yang digunakan adalah YOLOv8m, *dropout* sebesar 0,2, *batch size* 64, *learning rate* 0,0001, dan *optimizer* Adam. Rincian ini dapat dilihat pada **Tabel 4. 3**.

Tabel 4. 3: Kombinasi yang menghasilkan mAP terbaik

Model	Dropout	Batch	Learning Rate	Optimizer	mAP
yolov8m	0,2	64	0,0001	Adam	0,87916



Gambar 4. 25: Perbandingan optimizer

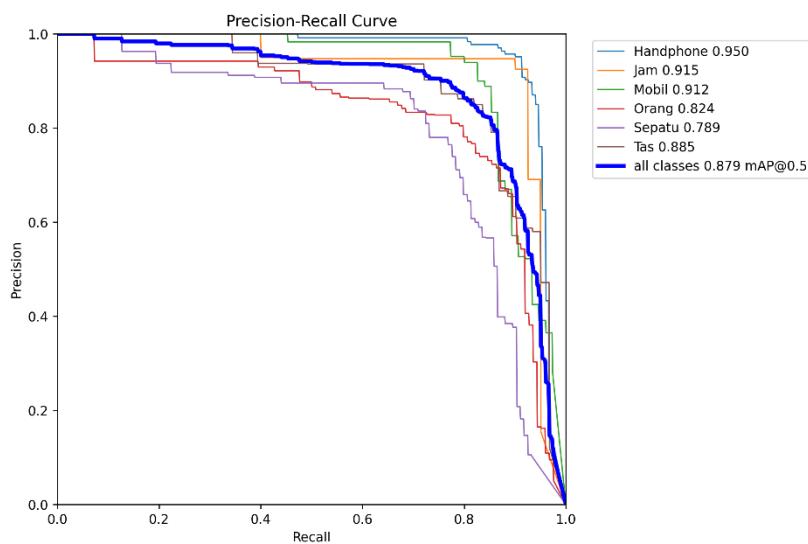
Selain itu, dapat dilihat bahwa *optimizer* Adam yang digunakan juga membantu memberikan hasil yang maksimal dibandingkan dengan *optimizer* RMSProp. Hal ini ditunjukkan dengan rata-rata mAP yang diberikan oleh *optimizer* Adam sebesar 0.8506 sedangkan rata-rata mAP yang diberikan oleh *optimizer* RMSProp adalah 0,6876. Hal ini pun dapat dilihat pada **Gambar 4. 25**.

```
model = '/kaggle/working/runs/detect/train/weights/best.pt'
data = '/kaggle/working/datasets/detect-2/data.yaml'

!yolo task=detect mode=val model=model data=data plots=True save_json=True
```

Gambar 4. 24: Code untuk melakukan validasi

Selanjutnya setelah mendapatkan model yang terbaik dari percobaan yang telah dilakukan, maka akan dilakukan validasi untuk melihat performa dari model tersebut. Hal ini juga dilakukan agar memastikan model tidak mengalami *overfitting* atau *underfitting* (Kumar dkk., 2021). Untuk melakukan validasi pada model, dapat menggunakan *code* pada **Gambar 4. 24**. Ketika *code* ini dijalankan maka akan menyimpan hasil *plot* dan hasil validasi dalam bentuk file .json. Hal ditunjukkan dengan parameter *plots* dan *save_json* yang diatur menjadi *true*.



Gambar 4. 26: Grafik precision dan recall

Validasi yang dilakukan akan menghasilkan beberapa grafik. Pada grafik di **Gambar 4. 26** sumbu X mewakili *recall*, yang mengukur proporsi positif yang benar-benar terdeteksi dari total positif sebenarnya dan sumbu Y mewakili *precision*, yang mengukur proporsi prediksi positif yang benar-benar benar dari total prediksi positif. *Precision-Recall Curve* yang baik akan mendekati sudut kiri atas grafik, yang berarti model memiliki *precision* dan *recall* yang tinggi secara bersamaan. Dari sini dapat dilihat bahwa kurva untuk kelas "Handphone" (berwarna biru muda) mendekati sudut kiri atas, menunjukkan bahwa model sangat baik dalam mendeteksi handphone dengan nilai *Average Precision* (AP) sebesar 0.950. Ini

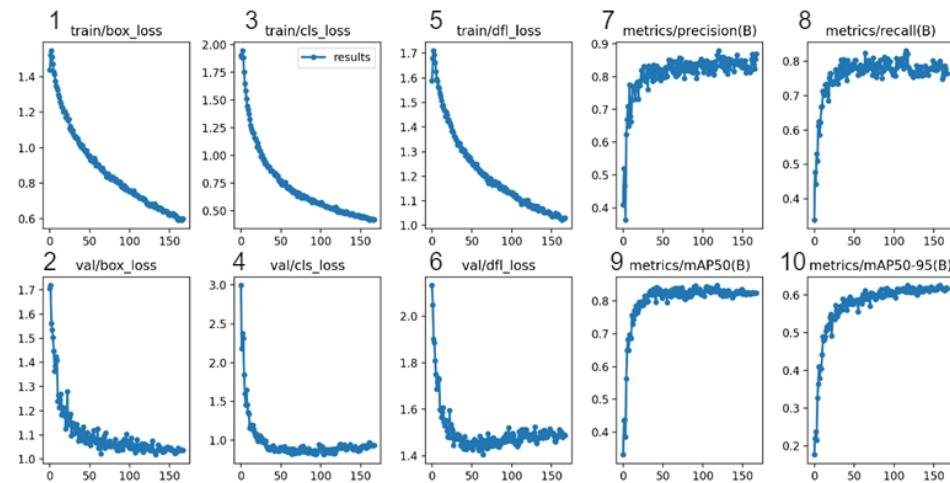
menunjukkan bahwa model mampu mendeteksi handphone dengan sangat akurat dan sedikit kesalahan. Sebaliknya, kurva untuk kelas "Sepatu" (berwarna merah) lebih rendah, dengan nilai AP sebesar 0.789, yang menunjukkan bahwa model kurang akurat dalam mendeteksi sepatu dibandingkan handphone. Kemudian kurva tebal berwarna hitam yang mewakili "*all classes*" menunjukkan kinerja rata-rata model untuk semua kelas, dengan nilai *mean Average Precision* (mAP) sebesar 0.879 pada *threshold* 0.5. Ini memberikan gambaran umum bahwa model memiliki kinerja yang baik secara keseluruhan, meskipun ada variasi dalam kinerja untuk masing-masing kelas. Sehingga secara keseluruhan, dengan kurva yang mendekati sudut kiri atas, kita dapat menyimpulkan bahwa model yang dihasilkan memiliki nilai *precision* dan *recall* yang bagus untuk sebagian besar kelas yang dianalisis. Ini menunjukkan bahwa model cukup andal dalam mendeteksi objek pada berbagai kategori dengan tingkat kesalahan yang rendah. (Jönsson Hyberg & Sjöberg, 2023; Terven & Cordova-Esparza, 2023).

PREDICTED	Handphone	0,94	0,05		0,02	0,11	
	Jam	0,01	0,95			0,03	
	Mobil		0,84			0,07	
	Orang			0,85		0,35	
	Sepatu		0,04		0,79	0,24	
	Tas				0,87	0,20	
	Background	0,05	0,05	0,07	0,15	0,21	0,11
		Handphone	Jam	Mobil	Orang	Sepatu	Tas
					TRUE		Background

Gambar 4. 27: Confusion matrix percobaan ketujuh

Berikut akan ditampilkan *confusion matrix* yang dapat dilihat pada **Gambar 4. 27**. Perlu diingat bahwa *confusion matrix* yang ditampilkan pada **Gambar 4. 27** adalah *confusion matrix* yang telah diubah agar lebih jelas dan untuk *confusion*

matrix yang asli ada pada **Lampiran 1**. Berdasarkan hasil *confusion matrix* tersebut, dapat dilihat misalnya, pada baris pertama dan kolom pertama (Handphone), nilai 0.94 menunjukkan bahwa 94% dari semua *instance* yang sebenarnya adalah Handphone diprediksi dengan benar sebagai Handphone. Demikian juga, pada baris pertama dan kolom ketiga, nilai 0.05 menunjukkan bahwa 5% dari *instance* yang sebenarnya Handphone salah diprediksi sebagai Jam. Secara keseluruhan, *confusion matrix* ini menunjukkan bahwa model memiliki performa yang baik untuk sebagian besar kelas, dengan tingkat akurasi yang tinggi. Namun, kelas sepatu memiliki tingkat kesalahan prediksi yang lebih tinggi di mana model sering dideteksi sebagai *background*. Hal inilah yang cukup menjelaskan kenapa AP pada objek sepatu di gambar 4.26 lebih kecil dibandingkan dengan objek-objek lainnya.



Gambar 4. 28: Kumpulan grafik hasil pelatihan model

Selanjutnya akan ditampilkan 10 grafik yang bertujuan untuk melihat hasil dari pelatihan dan evaluasi dari model tersebut. Secara rinci dari penjelasan grafik dapat dilihat dari **Tabel 4. 4**.

Tabel 4. 4: Tabel penjelasan dari grafik 4.28

Nomor Grafik	Penjelasan
Grafik Pertama	Grafik pertama menunjukkan "train/box_loss" yang mengindikasikan <i>loss</i> atau kerugian <i>bounding box</i> selama pelatihan. <i>Loss</i> ini berkurang secara konsisten dari sekitar 1,4 menjadi mendekati 0,1 saat jumlah iterasi pelatihan meningkat hingga 150. Penurunan yang signifikan ini menunjukkan bahwa model belajar untuk memprediksi <i>bounding box</i> objek dengan lebih akurat seiring berjalannya waktu.
Grafik Kedua	Grafik kedua adalah "val/box_loss" yang menunjukkan <i>loss</i> atau kerugian <i>bounding box</i> pada data validasi. Mirip dengan <i>loss</i> pelatihan, <i>loss</i> validasi ini juga menurun secara keseluruhan dari sekitar 1,6 menjadi mendekati 0,5, meskipun fluktuasi lebih besar terlihat di awal proses pelatihan. Ini menunjukkan bahwa model juga meningkatkan kinerjanya pada data yang tidak terlihat selama pelatihan.
Grafik Ketiga	Grafik ketiga menampilkan "train/cls_loss" yang mengindikasikan <i>loss</i> atau kerugian klasifikasi selama pelatihan. <i>Loss</i> ini berkurang dari sekitar 2,0 menjadi di bawah 0,2 setelah 150 iterasi pelatihan, menunjukkan bahwa model semakin akurat dalam mengklasifikasikan objek di dalam gambar.
Grafik Keempat	Grafik keempat adalah "val/cls_loss" yang menunjukkan <i>loss</i> atau kerugian klasifikasi pada data validasi. Grafik ini juga menurun dari sekitar 2,5 menjadi sekitar 0,5, meskipun dengan beberapa fluktuasi, yang menunjukkan bahwa model menjadi lebih baik dalam mengklasifikasikan objek pada data validasi seiring waktu
Grafik Kelima	Grafik kelima menunjukkan "train/dfl_loss" yang mungkin merujuk pada <i>loss</i> atau kerugian terkait distribusi atau fitur lainnya selama pelatihan. <i>Loss</i> ini berkurang dari sekitar 1,7 menjadi di bawah 0,1, menunjukkan peningkatan konsistensi atau kesesuaian fitur yang diukur selama pelatihan.
Grafik Keenam	Grafik keenam adalah "val/dfl_loss" yang menunjukkan <i>loss</i> atau kerugian yang sama pada data validasi. <i>Loss</i> ini juga menurun dari sekitar 1,4 menjadi sekitar 0,4, menunjukkan bahwa model terus meningkatkan kinerjanya dalam aspek ini pada data validasi.
Grafik Ketujuh	Grafik ketujuh menunjukkan "metrics/precision(B)" yang mengukur <i>precision</i> pada data validasi. Nilai <i>precision</i> ini meningkat secara cepat dari sekitar 0,2 menjadi stabil di sekitar 0,8 setelah sekitar 30 iterasi, menunjukkan bahwa model menjadi sangat akurat dalam mengidentifikasi objek yang benar di antara semua prediksi positifnya.
Grafik Kedelapan	Grafik kedelapan adalah "metrics/recall(B)" yang mengukur <i>recall</i> pada data validasi. <i>Recall</i> meningkat dari 0 hingga sekitar 0,75, menunjukkan bahwa model menjadi semakin mampu mengidentifikasi sebagian besar objek yang benar-benar ada dalam gambar.

Grafik Kesembilan	Grafik kesembilan menunjukkan "metrics/mAP50(B)" yang mengukur mean Average Precision pada IoU threshold 50%. Nilai ini meningkat tajam hingga sekitar 0,75, menunjukkan kinerja yang baik dalam mendeteksi dan mengklasifikasikan objek pada threshold tersebut.
Grafik Kesepuluh	Grafik kesepuluh adalah "metrics/mAP50-95(B)" yang mengukur mean Average Precision pada berbagai IoU threshold dari 50% hingga 95%. Nilai ini juga meningkat secara bertahap hingga sekitar 0,5, menunjukkan kemampuan model untuk tetap akurat pada berbagai threshold overlap yang lebih ketat.

Secara keseluruhan, grafik-grafik ini menunjukkan bahwa model mengalami peningkatan performa yang signifikan baik dalam pelatihan maupun validasi, yang ditandai dengan penurunan grafik *loss* dan peningkatan metrik evaluasi. Kombinasi dari penurunan nilai *loss*, nilai *precision*, *recall* dan nilai mAP yang tinggi menunjukkan bahwa model mampu mendeteksi objek dengan akurat dan efisien dan generalisasi yang baik.

```

test_path = "test-image"
model = YOLO(f"model-result/model-terbaik/best.pt")

results = model.predict(source=test_path)
names = model.names

for i, result in enumerate(results):
    boxes = result.boxes
    labels = result.names

    detected_labels = []

    for box in boxes:
        label = names[int(box.cls)]
        detected_labels.append(label)

    print(f"Detected labels for image {i+1}: {detected_labels}")
    res_plotted = result.plot()
    res_plotted_rgb = cv2.cvtColor(res_plotted, cv2.COLOR_BGR2RGB)

    plt.figure(figsize=(10, 10))
    plt.imshow(res_plotted_rgb)
    plt.axis('off')
    plt.show()

```

Gambar 4. 29: Code untuk menjalankan testing

Setelah dilakukan validasi untuk melihat performa dari model terbaik, selanjutnya akan dilakukan testing untuk melihat kemampuannya untuk

digeneralisasi pada data baru yang belum pernah dilihat sebelumnya. Hal ini dilakukan juga agar memberikan gambaran yang lebih realistik tentang bagaimana model akan bekerja di lingkungan produksi atau pada data yang sebenarnya. Untuk testing ini akan dilakukan dengan data gambar yang berjumlah 142 gambar. *Code* untuk testing akan menggunakan **Gambar 4. 29**.

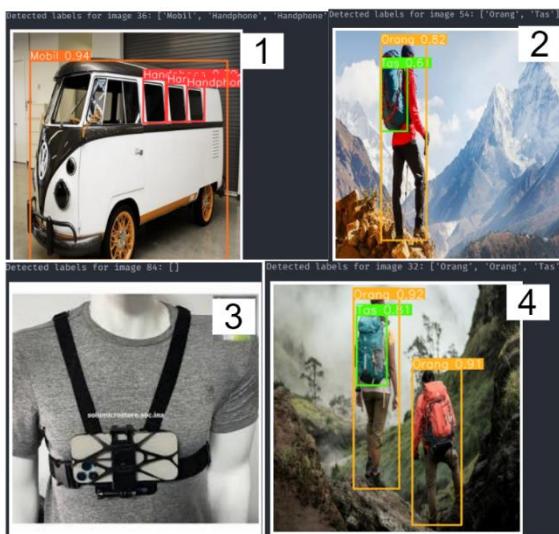
Code tersebut melakukan prediksi objek pada gambar-gambar yang terletak di `test_path` menggunakan model yang telah dilatih sebelumnya. Pertama, program mengambil daftar label kelas yang dikenali oleh model. Selanjutnya, untuk setiap gambar di `test_path`, program memproses hasil prediksi dengan mengambil nilai *bounding box* yang menandai lokasi objek yang terdeteksi. Dari setiap *bounding box* yang ditemukan, program mengambil nama kelas objek dengan mencocokkan indeks kelas dari *bounding box* dengan daftar nama kelas yang telah diambil sebelumnya. Setelah itu, program mencetak nama kelas yang terdeteksi untuk setiap gambar. Terakhir, program menampilkan gambar-gambar tersebut dengan *bounding box* yang menandai posisi objek yang terdeteksi. Gambar-gambar hasil prediksi tersebut ditampilkan tanpa *axis* untuk memperjelas hasil deteksi objek.



Gambar 4. 30: Gambar yang diprediksi dengan benar

Dengan demikian, program ini secara keseluruhan melakukan prediksi objek, mengambil label hasil prediksi, dan menampilkan gambar dengan objek yang terdeteksi beserta labelnya.

Dari 142 gambar ini sebagian besar gambar dapat dideteksi dengan benar, namun ada beberapa juga gambar yang bisa dideteksi dengan kurang benar oleh model. Seperti yang terlihat pada **Gambar 4. 30**, model dapat mendeteksi objek pada gambar dengan benar selama gambar yang diprediksi memiliki kualitas yang bagus atau gambar yang jelas. Dengan kualitas yang kurang bagus atau dengan objek yang kurang terlihat jelas mampu membuat model salah atau tidak dapat memprediksi objek seperti yang ada **Gambar 4. 31** khususnya pada gambar 3. Selain itu, terlihat bahwa model masih mendeteksi objek yang memiliki bentuk atau fitur yang serupa seperti pada gambar mobil pada nomor 1. Kemudian model pun kurang mampu mendeteksi sepatu yang sering dianggap sebagai *background* seperti yang ditunjukkan pada gambar 2 dan gambar 4.



Gambar 4. 31: Gambar yang kurang tepat diprediksi

Berdasarkan hasil pelatihan untuk mendapatkan model yang baik, validasi dan testing terhadap model, maka dapat diputuskan bahwa pada skenario atau

percobaan ketujuh akan digunakan dalam *website* deteksi objek sebagai bentuk implementasi hasil akhir karena model pada percobaan ketujuh merupakan model yang baik dibandingkan dengan model yang lain.

4.6. Implementasi Aplikasi Deteksi Objek

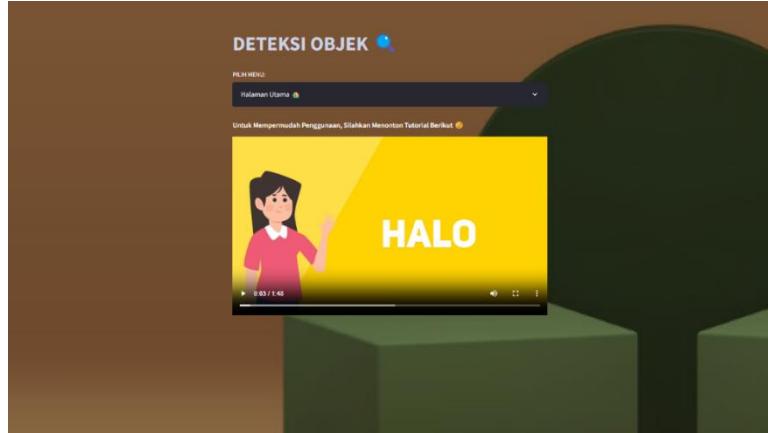
4.6.1. Pengguna User Interface

Aplikasi ini dirancang untuk digunakan oleh beberapa kelompok pengguna, yaitu orang tua yang memiliki anak usia 2-4 tahun, guru PAUD, dan masyarakat umum di mana sejalan dengan target dari kuesioner yang dibuat pada poin 3.9.1. Karena target aplikasi ini merupakan orang tua, Guru PAUD, dan masyarakat umum, maka tampilan antarmuka akan dirancang agar lebih ramah dan dengan desain yang sederhana. Selain itu juga memiliki antarmuka yang mudah dipahami dan informasi lengkap tentang penggunaan aplikasi menjadi penting, dengan akses mudah ke panduan pengguna atau pusat bantuan jika diperlukan.

4.6.2. Interface Aplikasi

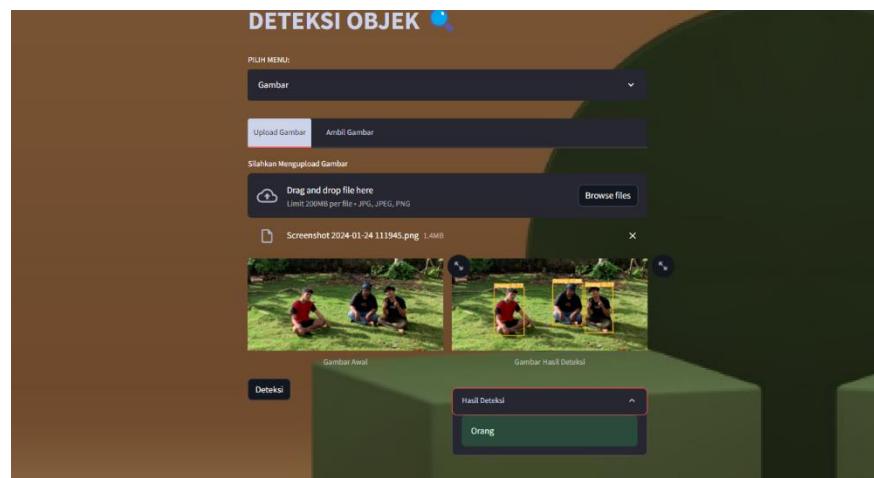
Implementasi model ke dalam *interface* akan dibantu dengan *framework* Streamlit. Alasan mengapa menggunakan *framework* ini adalah karena pihak Streamlit telah menyediakan *cloud* khusus bagi para anggota komunitas untuk *deploy* hasil kerjanya (Sholahuddin dkk., 2023). Pada *website* ini akan terdapat beberapa fitur utama. Fitur-fitur tersebut antara lain mendeteksi gambar dengan mengunggah gambar atau dengan mengambilnya secara langsung, mendeteksi video dengan mengunggah video atau dengan beberapa video yang sudah disiapkan sebelumnya, mendeteksi dengan memasukan *link* YouTube, dan mendeteksi secara *realtime*. Pengimplementasian ini akan melibatkan beberapa *function* untuk mempermudah proses deteksi dengan model yang sudah dibuat sebelumnya.

Pada **Gambar 4. 32** adalah *interface* yang dibuat di Streamlit. Ketika pertama kali diakses. Pada halaman ini akan menampilkan *dropdown* dari menu-menu yang disediakan juga menampilkan video tutorial penggunaan *website*.



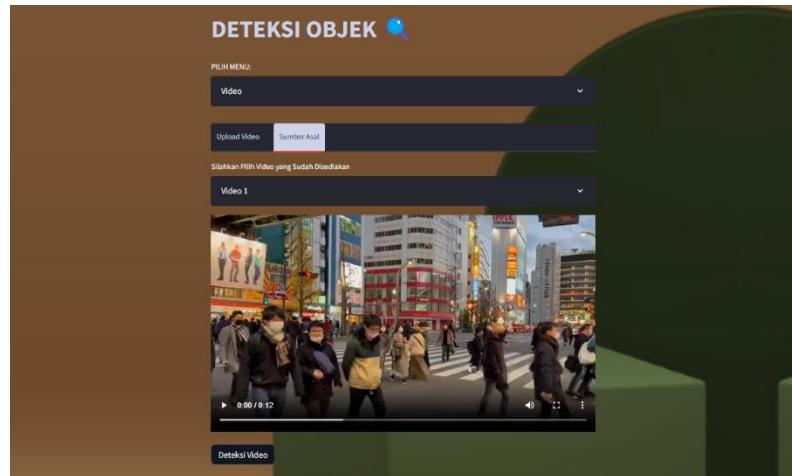
Gambar 4. 32: Tampilan halaman pertama ketika website diakses

Berikutnya, tampilan pada **Gambar 4. 33** adalah tampilan jika memilih menu gambar dan telah melakukan deteksi dengan mengunggah foto atau gambar. Dapat dilihat bahwa di bawah *dropdown* menu terdapat *tab* untuk memilih ingin menggunakan fitur yang sama. Hal ini pun berlaku pada tampilan jika memilih menu video seperti yang terlihat pada **Gambar 4. 34**.



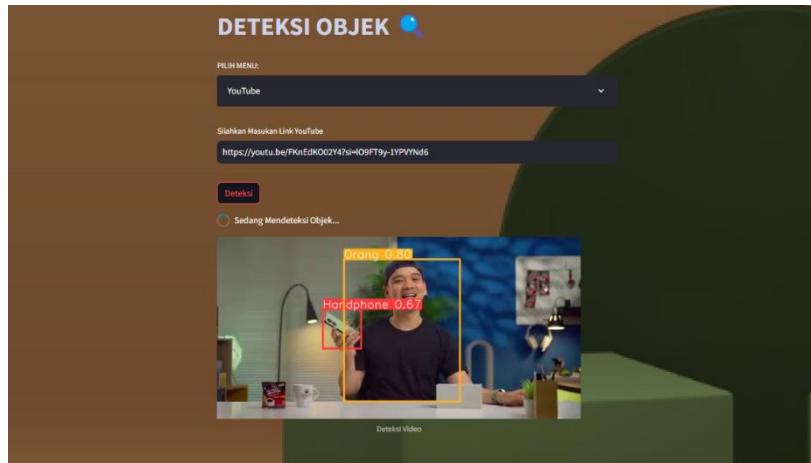
Gambar 4. 33: Tampilan halaman jika memilih menu gambar

Pada menu ini telah disediakan lima video yang dapat langsung digunakan untuk deteksi sehingga user tidak perlu mengunggah videonya sendiri.



Gambar 4. 34: Tampilan jika memilih menu video

Selanjutnya pada **Gambar 4. 35** dapat dilihat tampilan jika memilih menu youtube. Pada menu ini, user hanya perlu memasukan link YouTube yang sudah disalin dan menekan tombol ‘deteksi’.



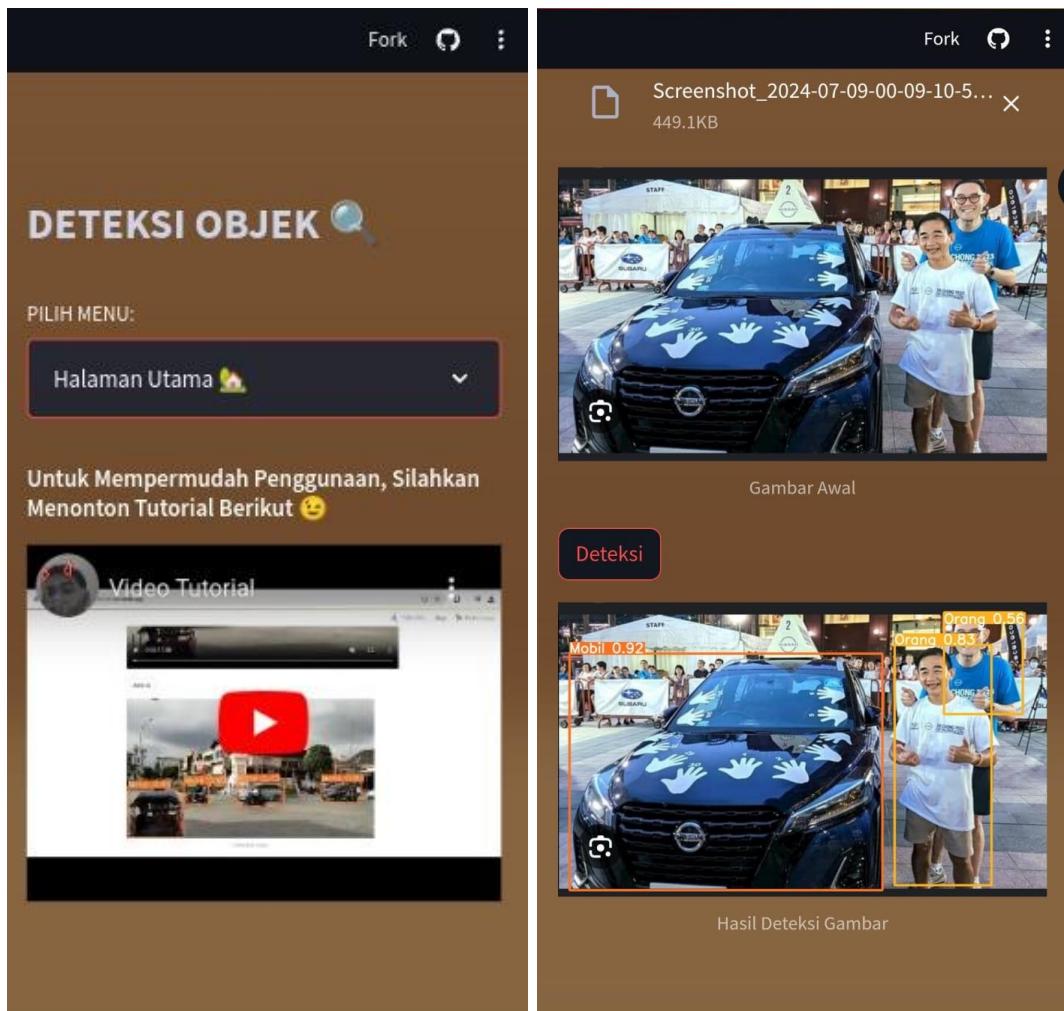
Gambar 4. 35: Tampilan jika memilih menu youtube

Terakhir adalah tampilan jika user memilih menu Real-Time. Seperti yang terlihat pada **Gambar 4. 37**, menu ini akan mendekripsi secara realtime menggunakan kamera baik ponsel maupun komputer.



Gambar 4. 37: Tampilan ketika memilih menu Real-Time

Berikut adalah tampilan jika *website* digunakan melalui ponsel untuk tampilan awal ketika diakses dan halaman hasil deteksi menggunakan menu upload gambar.



Gambar 4. 36: Tampilan jika menggunakan atau ponsel

4.6.3. Implementasi Code untuk Deteksi

Terdapat beberapa *function* yang diimplementasi untuk menunjang proses deteksi di tiap menu.

1. Function untuk menampilkan hasil deteksi

Fungsi `showDetectFrame` merupakan *function* yang menggabungkan deteksi objek dalam gambar dengan pembuatan audio berbasis teks yang menggambarkan objek-objek terdeteksi tersebut. Fungsi ini menerima parameter berupa tingkat kepercayaan (*conf*), model deteksi (*model*), komponen tampilan streamlit (*st_frame*), gambar (*image*), dan teks keterangan opsional (*caption*). Fungsi dimulai dengan memprediksi objek-objek dalam gambar menggunakan model deteksi dengan tingkat kepercayaan yang diberikan. Hasil prediksi mencakup kotak-kotak deteksi (*boxes*) dan label-label nama objek yang terdeteksi. Selanjutnya, gambar hasil deteksi dengan kotak-kotak dan label-label tersebut digambar ulang dan ditampilkan menggunakan streamlit. Fungsi ini juga mengumpulkan semua label yang terdeteksi ke dalam sebuah set untuk menghilangkan duplikasi, dan kemudian mencetak label-label tersebut.

Bagian penting berikutnya adalah fungsi internal `get_audio_bytes`, yang bertugas mengubah daftar label terdeteksi menjadi teks yang kemudian dikonversi menjadi audio menggunakan Google Text-to-Speech (gTTS) dalam bahasa Indonesia. Jika tidak ada objek yang terdeteksi, teks default yang dihasilkan adalah "Tidak ada

objek yang terdeteksi". Audio yang dihasilkan kemudian disimpan dalam buffer dan dikonversi menjadi format base64 untuk memungkinkan *embed* audio dalam format HTML. Terakhir, audio tersebut diputar secara otomatis di halaman web menggunakan komponen HTML yang *diembed* di dalam streamlit. Fungsi ini secara keseluruhan menciptakan pengalaman interaktif yang tidak hanya menampilkan hasil deteksi visual, tetapi juga menyertakan informasi audio tentang apa yang terdeteksi dalam gambar. Untuk *capture code* dapat dilihat pada **Lampiran 2**.

2. Function untuk menjalankan menu YouTube

Fungsi `play_youtube` memungkinkan pengguna untuk mendeteksi objek dalam video YouTube melalui *link* yang diberikan. Fungsi ini dimulai dengan meminta pengguna untuk memasukkan *link* YouTube melalui kotak teks yang disediakan oleh streamlit. Setelah *link* dimasukkan dan tombol "Deteksi" ditekan, fungsi menampilkan sebuah *spinner* sebagai indikasi bahwa proses deteksi sedang berlangsung. Dalam blok *try*, fungsi menggunakan pustaka `pytube` untuk mengunduh video YouTube dengan resolusi 720p dalam format MP4. Video tersebut kemudian dibaca *frame* per *frame* menggunakan OpenCV (`cv2.VideoCapture`). Setiap *frame* video diproses oleh fungsi `showDetectFrame` yang mendeteksi objek dalam *frame* tersebut menggunakan model deteksi objek yang diberikan dan menampilkan hasil deteksi dalam komponen streamlit yang disediakan. Proses ini berlangsung selama video masih dapat dibaca frame-nya. Jika terjadi

kesalahan selama proses, pesan *error* akan ditampilkan kepada pengguna. Fungsi ini menggabungkan kemampuan untuk mengunduh dan memproses video YouTube dengan deteksi objek secara real-time, memberikan pengalaman interaktif dan informatif kepada pengguna.

Untuk *capture code* dapat dilihat pada **Lampiran 3**.

3. Function untuk menjalankan menu realtime

Fungsi live mengimplementasikan deteksi objek secara *real-time* menggunakan kamera melalui streaming WebRTC. Fungsi ini mengatur konfigurasi WebRTC dengan parameter yang diperlukan untuk menghubungkan ke server ICE (*Interactive Connectivity Establishment*) menggunakan transportasi *relay*, dan kemudian memulai *streaming* dengan *transformer* video khusus yang didefinisikan oleh kelas VideoTransformer. Kelas VideoTransformer mewarisi dari VideoTransformerBase dan berfungsi untuk menangani proses deteksi objek pada setiap *frame* video yang diterima. Pada inisialisasi, kelas ini menerima model deteksi objek dan tingkat kepercayaan sebagai parameter, serta menyimpan set label objek yang terakhir terdeteksi. Metode *transform* mengonversi *frame* video menjadi format numpy array, menjalankan deteksi objek pada *frame* tersebut menggunakan model yang diberikan, dan menghasilkan gambar dengan anotasi deteksi. Jika terdapat perubahan dalam label yang terdeteksi dibandingkan dengan *frame* sebelumnya, metode ini akan memicu fungsi speak_labels dalam thread terpisah untuk menghindari *blocking*. Fungsi speak_labels mengubah label terdeteksi menjadi teks dan

kemudian menggunakan Google Text-to-Speech (gTTS) untuk mengonversi teks tersebut menjadi audio. Audio ini kemudian diputar menggunakan pustaka pygame, memberikan umpan balik suara secara *real-time* mengenai objek yang terdeteksi di dalam *frame* video. Kombinasi dari fungsi ini memberikan pengalaman deteksi objek secara live dengan informasi visual dan audio, meningkatkan interaktivitas dan kegunaan aplikasi. Untuk *capture code* dapat dilihat pada **Lampiran 4** dan **Lampiran 5**.

4. Function untuk menjalankan menu upload video

Fungsi `process_uploaded_video` memungkinkan pengguna untuk mengunggah dan mendeteksi objek dalam video yang diunggah melalui antarmuka Streamlit. Fungsi ini dimulai dengan menampilkan komponen *file uploader* yang menerima *file* video dengan ekstensi MP4, AVI, atau MOV. Ketika pengguna mengunggah video, fungsi menulis konten video tersebut ke *file* sementara menggunakan `NamedTemporaryFile` untuk penyimpanan sementara. Video yang diunggah kemudian dibaca kembali dan ditampilkan menggunakan komponen `st.video` dari Streamlit untuk pratinjau. Jika pengguna menekan tombol "Deteksi", fungsi memulai proses deteksi objek dengan menampilkan *spinner* sebagai indikasi proses sedang berjalan. Video yang diunggah dibuka menggunakan OpenCV (`cv2.VideoCapture`), dan *frame* video dibaca satu per satu. Setiap *frame* diproses oleh fungsi `showDetectFrame`, yang mendeteksi objek dalam *frame* tersebut dan menampilkan hasilnya pada komponen Streamlit. Proses ini

berlangsung hingga semua *frame* video diproses atau hingga terjadi kesalahan, yang akan ditampilkan sebagai pesan *error*. Fungsi ini memberikan kemampuan interaktif kepada pengguna untuk mengunggah video dan melihat hasil deteksi objek secara visual dalam setiap *frame* video yang diunggah. Untuk *capture code* dapat dilihat pada **Lampiran 6**.

5. Function untuk menjalankan menu video asal

Fungsi `play_stored_video` dirancang untuk memungkinkan pengguna memilih dan mendeteksi objek dalam video yang sudah disediakan melalui antarmuka Streamlit. Fungsi ini dimulai dengan menampilkan *dropdown* menu (selectbox) yang berisi daftar nama video yang tersedia, diambil dari *dictionary settings.VIDEOS_DICT*. Setelah pengguna memilih video, fungsi membaca konten video dari *path* yang sesuai dalam *dictionary* dan menampilkannya menggunakan komponen `st.video` dari Streamlit untuk pratinjau. Jika pengguna menekan tombol "Deteksi Video", fungsi memulai proses deteksi objek dengan menampilkan *spinner* sebagai indikasi proses sedang berlangsung. Video yang dipilih kemudian dibuka menggunakan OpenCV (`cv2.VideoCapture`), dan *frame* video dibaca satu per satu. Setiap *frame* diproses oleh fungsi `showDetectFrame`, yang mendeteksi objek dalam *frame* tersebut dan menampilkan hasilnya pada komponen Streamlit. Proses ini terus berlangsung hingga semua *frame* video diproses atau hingga terjadi kesalahan, yang kemudian akan ditampilkan sebagai pesan *error*. Fungsi ini memberikan cara mudah bagi pengguna untuk

memilih dari video yang sudah disediakan dan melihat hasil deteksi objek dalam setiap *frame* video yang dipilih. Untuk *capture code* dapat dilihat pada **Lampiran 7**.

6. Function untuk menjalankan menu pengambilan foto

Fungsi `take_picture` dirancang untuk memungkinkan pengguna mengambil gambar menggunakan kamera melalui antarmuka Streamlit dan mendeteksi objek dalam gambar tersebut. Fungsi dimulai dengan menampilkan komponen kamera yang memungkinkan pengguna untuk mengambil gambar. Setelah gambar diambil, gambar tersebut disimpan sementara sebagai *file .JPG* menggunakan `NamedTemporaryFile`. Jika pengguna menekan tombol "Deteksi Foto", fungsi memulai proses deteksi objek dengan menampilkan *spinner* sebagai indikasi proses sedang berjalan. Gambar yang diambil dibuka menggunakan OpenCV (`cv2.VideoCapture`). *Frame* diproses oleh fungsi `showDetectFrame`, yang mendeteksi objek dalam *frame* tersebut dan menampilkan hasilnya pada komponen Streamlit. Fungsi ini menyediakan cara interaktif bagi pengguna untuk mengambil gambar secara langsung dan melihat hasil deteksi objek dalam gambar tersebut. Untuk *capture code* dapat dilihat pada **Lampiran 8**.

7. Function untuk menjalankan menu upload foto

Fungsi `up_picture` memungkinkan pengguna mengunggah gambar untuk dideteksi objeknya melalui antarmuka Streamlit. Fungsi dimulai dengan menampilkan komponen `file_uploader` yang memungkinkan pengguna mengunggah gambar dengan format *JPG*,

JPEG, atau PNG. Jika ada gambar yang diunggah, gambar tersebut akan ditampilkan dalam kolom pertama dengan keterangan "Gambar Awal". Jika pengguna menekan tombol "Deteksi", proses deteksi objek akan dimulai dengan menampilkan *spinner* sebagai indikasi proses sedang berjalan. Gambar yang diunggah akan diproses oleh fungsi showDetectFrame, yang mendeteksi objek dalam gambar dan menampilkan hasilnya di kolom kedua dengan keterangan "Hasil Deteksi Gambar". Jika tidak ada gambar yang diunggah, fungsi akan menampilkan gambar *default* di kedua kolom: satu sebagai "Gambar Awal" dan satu lagi sebagai "Hasil Deteksi". Jika terjadi kesalahan saat membaca *file*, pesan *error* akan ditampilkan. Fungsi ini memberikan pengalaman interaktif kepada pengguna untuk mengunggah gambar dan melihat hasil deteksi objek secara visual dalam gambar yang diunggah atau gambar *default*. Untuk *capture code* dapat dilihat pada **Lampiran 9**.

4.6.4. Alur atau Cara Kerja Aplikasi

Aplikasi yang telah dibuat dan di masukan ke dalam *cloud* dari Streamlit, dapat diakses melalui link <https://thesis-detec-result.streamlit.app>. Salanjutnya terkait dengan alur atau cara kerja dari aplikasi yang dibuat, akan dijelaskan menggunakan narasi *use case* untuk tiap menu seperti pada tabel-tabel di bawah ini.

1. Narasi use case akses awal

Tabel 4. 5: Tabel narasi use case akses awal

Use Case Akses Awal	
Aktor	Orang tua, guru PAUD, dan masyarakat umum

Deskripsi	Use case ini menggambarkan aktor yang mengakses aplikasi.
Kondisi awal	Aktor mengakses aplikasi melalui link
Kondisi akhir	Aktor dapat menggunakan aplikasi
Skenario Use Case	
Aksi Aktor	Reaksi Sistem
Step 1: Aktor mengakses aplikasi melalui link yang telah disiapkan	Step 2: Sistem menampilkan halaman utama sistem ketika diakses pertama kali.
Step 3a: Aktor memutar video yang telah disiapkan untuk menonton tutorial penggunaan	Step 4a: Sistem memutar video dari YouTube yang telah disediakan
Step 3b: Aktor menekan <i>dropdown</i> menu	Step 4b: Sistem menampilkan daftar menu yang ada di <i>dropdown</i>

2. Narasi use case menu gambar

Tabel 4. 6: Tabel narasi use case menu gambar

Use Case Menu Gambar	
Aktor	Orang tua, guru PAUD, dan masyarakat umum
Deskripsi	Use case ini menggambarkan aktor yang mengakses menu gambar.
Kondisi awal	Aktor berada pada halaman utama
Kondisi akhir	Aktor dapat menggunakan menu gambar untuk memprediksi gambar.
Skenario Use Case	
Aksi Aktor	Reaksi Sistem
Step 1:	

Aktor menekan <i>dropdown</i> menu dan memilih menu gambar Step 3a (tab upload gambar): Aktor memasukkan gambar dengan menekan tombol ‘Browser File’ atau dengan aksi <i>drag and drop</i>	Step 2: Sistem menampilkan halaman menu gambar Step 4a: Sistem mendapatkan gambar yang dimasukkan kemudian menampilkan gambar tersebut sebagai ‘Gambar Awal’
Step 5a: Aktor menekan tombol ‘Deteksi’	Step 6a: Sistem menampilkan <i>spinner</i> sebagai reaksi dan memprediksi gambar yang telah dimasukkan kemudian menampilkan gambar hasil prediksi dan memutar audio hasil deteksi.
Step 3b (tab ambil gambar): Aktor memilih tab ‘Ambil Gambar’	Step 4b: Sistem menampilkan halaman untuk mengambil gambar sekaligus menyalakan kamera yang sudah diberi izin sebelumnya.
Step 4b: Aktor mengambil atau mengcapture gambar dari kamera yang dibuka oleh sistem.	Step 5b: Sistem membaca gambar hasil <i>capture</i> kemudian menampilkan tombol ‘Deteksi’
Step 6b: Aktor menekan tombol ‘Deteksi’	Step 7b: Sistem menampilkan <i>spinner</i> sebagai reaksi dan memprediksi gambar yang telah dimasukkan kemudian menampilkan gambar hasil prediksi dan memutar audio hasil deteksi.

3. Narasi use case menu video

Tabel 4. 7: Tabel narasi use case menu video

Use Case Menu Video	
Aktor	Orang tua, guru PAUD, dan masyarakat umum

Deskripsi	Use case ini menggambarkan aktor yang mengakses menu video.
Kondisi awal	Aktor berada pada halaman utama
Kondisi akhir	Aktor dapat menggunakan menu video untuk memprediksi video.
Skenario Use Case	
Aksi Aktor	Reaksi Sistem
<p>Step 1: Aktor menekan <i>dropdown</i> menu dan memilih menu video</p> <p>Step 3a (tab upload video): Aktor memasukkan video dengan menekan tombol ‘Browser File’ atau dengan aksi <i>drag and drop</i></p> <p>Step 5a: Aktor menekan tombol ‘Deteksi’</p> <p>Step 3b (tab ambil gambar): Aktor memilih tab ‘Sumber Asal’</p> <p>Step 5b: Aktor menekan tombol ‘Deteksi’</p>	<p>Step 2: Sistem menampilkan halaman menu video</p> <p>Step 4a: Sistem mendapatkan video yang dimasukkan kemudian menampilkan video tersebut sebagai ‘Video Awal’</p> <p>Step 6a: Sistem menampilkan <i>spinner</i> sebagai reaksi dan memprediksi video yang telah dimasukkan kemudian menampilkan video hasil prediksi dan memutar audio hasil deteksi.</p> <p>Step 4b: Sistem menampilkan <i>dropdown</i> yang berisi daftar video yang sudah dimasukkan ke dalam sistem. Sebagai nilai awal, sistem akan menampilkan video 1 dari daftar.</p> <p>Step 6b: Sistem menampilkan <i>spinner</i> sebagai reaksi dan memprediksi video yang telah dipilih kemudian menampilkan gambar hasil prediksi dan memutar audio hasil deteksi.</p>

4. Narasi use case menu YouTube

Tabel 4. 8: Tabel narasi use case menu video

Use Case Menu Video	
Aktor	Orang tua, guru PAUD, dan masyarakat umum
Deskripsi	Use case ini menggambarkan aktor yang mengakses menu YouTube.
Kondisi awal	Aktor berada pada halaman utama
Kondisi akhir	Aktor dapat menggunakan menu YouTube untuk memprediksi video.
Skenario Use Case	
Aksi Aktor	
Step 1: Aktor menekan <i>dropdown</i> menu dan memilih menu YouTube	Step 2: Sistem menampilkan halaman menu YouTube
Step 3: Aktor memasukkan <i>link</i> YouTube ke tempat yang sudah disiapkan dan menekan tombol ‘Deteksi’	Step 4: Sistem mengambil video di YouTube sesuai dengan <i>link</i> yang didapatkan kemudian sistem menampilkan <i>spinner</i> sebagai reaksi dan memprediksi video yang telah dimasukkan kemudian menampilkan video hasil prediksi dan memutar audio hasil deteksi.

5. Narasi use case real-time

Tabel 4. 9: Tabel narasi use case real-time

Use Case Menu Real-Time	
Aktor	Orang tua, guru PAUD, dan masyarakat umum
Deskripsi	Use case ini menggambarkan aktor yang mengakses menu <i>real-time</i> .
Kondisi awal	Aktor berada pada halaman utama
Kondisi akhir	Aktor dapat menggunakan menu <i>real-time</i> untuk memprediksi secara <i>real-time</i>
Skenario Use Case	
Aksi Aktor	
Step 1:	

<p>Aktor menekan <i>dropdown</i> menu dan memilih menu Real-Time</p> <p>Step 3 (opsional): Aktor memilih opsi ‘Select Device’</p> <p>Step 5 (opsional): Aktor menekan tombol ‘Done’</p> <p>Step 3: Aktor menekan tombol ‘Start’</p>	<p>Step 2: Sistem menampilkan halaman menu <i>real-time</i></p> <p>Step 4 (opsional): Sistem menampilkan opsi <i>device</i> yang ada di perangkat.</p> <p>Step 4: Sistem akan menyalakan kamera dari perangkat sekaligus menyinkronkan dengan hasil deteksi dan memutar suara dari hasil deteksi.</p>
---	---

Dengan aplikasi yang dibuat dengan alur yang ada, dapat dipastikan bahwa

aplikasi dapat dijalankan dengan baik untuk mendeteksi.

4.6.5. Manfaat Dari Aplikasi

Adapun manfaat atau *benefit* yang didapatkan jika menggunakan aplikasi yang telah dibuat adalah:

1. Mendukung proses perkembangan kognitif anak.

Aplikasi ini dapat merangsang keterampilan kognitif seperti pengenalan objek, memori visual, dan asosiasi kata-gambar.

2. Membantu orang tua dalam mengajarkan penggunaan perangkat lunak yang edukatif.

Orang tua dapat menggunakan aplikasi ini sebagai alat bantu untuk mengajarkan anak tentang teknologi dengan cara yang positif dan edukatif.

- Meningkatkan kualitas interaksi dan hubungan antara orang tua dan anak.

Dengan menggunakan aplikasi ini bersama, orang tua dan anak dapat memiliki waktu berkualitas yang bermakna.

- Meningkatkan kemampuan bahasa dan komunikasi.

Melalui pengenalan objek dan kata-kata baru, anak-anak dapat memperluas kosa kata dan kemampuan bahasa mereka.

4.7. Pengujian Kegunaan Aplikasi

4.7.1. Uji Validitas

Uji validitas akan dilakukan sebelum kuesioner akan disebarluaskan secara resmi. Hal ini dilakukan untuk mengetahui kelayakan sebuah instrumen yang akan diuji. Pada pengujian ini, kuesioner dan *website* akan disebarluaskan secara tertutup dengan mengirimkan *link* melalui media sosial WhatsApp kepada kenalan peneliti.

Adapun hasil dari kuesioner pengujian dari 30 responden dapat dilihat pada **Tabel 4. 10** dan untuk daftar pertanyaan dari kuesioner ini dapat dilihat di **Lampiran 10**.

Tabel 4. 10: Hasil kuesioner pengujian

R	S	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	TOTAL
R1	OT	4	4	4	4	3	4	4	3	4	3	3	40
R2	G	4	4	3	4	3	3	3	3	4	3	4	38
R3	G	4	4	4	3	4	4	4	4	4	4	4	43
R4	OT	4	4	3	4	4	4	4	4	4	3	4	42
R5	MU	3	3	3	3	4	3	3	3	3	3	3	34
R6	OT	3	3	3	3	3	3	3	3	3	3	3	33
R7	G	4	3	4	4	4	4	4	4	4	4	4	43
R8	G	4	4	3	3	2	4	4	4	3	3	3	37
R9	OT	4	4	4	4	4	4	4	4	4	4	4	44
R10	OT	4	4	4	3	4	3	4	3	4	3	4	40
R11	OT	4	4	4	3	4	4	4	4	4	3	4	42
R12	OT	4	4	3	4	4	4	4	4	4	4	4	43
R13	MU	3	3	3	3	3	3	3	3	3	3	3	33
R14	G	4	4	4	3	4	3	4	3	4	4	4	41
R15	G	4	4	3	4	4	4	4	4	4	3	4	42
R16	OT	3	3	3	3	4	3	3	3	3	3	3	34
R17	OT	4	4	3	3	2	4	4	4	3	3	3	37

R18	OT	4	4	4	4	4	4	4	4	3	4	43
R19	G	4	4	3	4	3	3	3	3	4	4	39
R20	OT	4	4	3	3	4	4	4	4	3	3	39
R21	MU	4	3	4	3	4	4	4	4	3	4	40
R22	MU	4	4	4	4	4	4	4	4	4	4	44
R23	MU	4	4	4	3	4	4	4	4	4	3	41
R24	G	4	3	4	4	4	4	4	4	4	3	42
R25	OT	4	4	3	3	4	3	3	4	4	3	39
R26	OT	4	4	4	3	3	3	4	3	4	4	40
R27	G	3	4	3	4	4	3	4	3	3	4	39
R28	OT	4	4	4	3	4	4	3	4	4	3	41
R29	OT	4	3	4	4	4	3	3	4	4	3	40
R30	G	3	4	4	4	4	4	4	3	4	3	41

Dari **Tabel 4. 10** tersebut, selanjutnya akan dilakukan perhitungan untuk mencari nilai *rHitung* tiap pertanyaan sesuai dengan rumus pada persamaan 1. Hasil perhitungan tersebut dapat dilihat pada **Tabel 4. 11**.

Tabel 4. 11: Hasil perhitungan rHitung

rHitung											
0,67	0,48	0,61	0,50	0,49	0,61	0,64	0,57	0,76	0,41	0,69	

Perhitungan nilai *rHitung* ini menggunakan bantuan Excel dengan menggunakan rumus = CORREL($\sum Q \sum T$). Selanjutnya untuk menentukan valid dan tidak validnya sebuah instrumen, nilai *rHitung* akan dibandingkan dengan nilai *rTabel* dengan distribusi signifikan sebesar 5%. Dikarenakan responden berjumlah 30 maka *rTabel* akan bernilai 0,349. Pada perbandingan ini, jika nilai *rHitung* lebih besar dari *rTabel* maka instrumen dikatakan valid, dan jika sebaliknya maka instrumen dikatakan tidak valid.

Secara lebih lanjut, hasil dari perbandingan tersebut dapat dilihat dari **Tabel 4. 12.**

Tabel 4. 12: Hasil uji validitas

rH	0,67	0,48	0,61	0,50	0,49	0,61	0,64	0,57	0,76	0,41	0,69
rT	0,349	0,349	0,349	0,349	0,349	0,349	0,349	0,349	0,349	0,349	0,349
K	V	V	V	V	V	V	V	V	V	V	V

Keterangan:

rH : *rHitung*

rT : *rTabel*

K : Keterangan

V : Valid

TV : Tidak Valid

Dari tabel tersebut dapat dilihat bahwa tiap instrumen dikatakan valid dan selanjutnya akan dilakukan pengujian reliabilitas.

4.7.2. Uji Reliabilitas

Setelah melakukan uji validitas untuk melihat apakah tiap instrumen valid untuk diujikan, selanjutnya akan dilakukan uji reliabilitas untuk melihat apakah instrumen dapat konsisten dalam mengukur hal yang harus diukur. Untuk menghitung suatu reliabilitas akan menggunakan rumus pada persamaan 2. Dengan mengacu pada **Tabel 3. 4** tentang tingkatan atau interval reliabilitas, maka akan dicari nilai r .

$$r = \left(\frac{11}{11-1} \right) \left(1 - \frac{2.641}{9.752} \right) \quad (7)$$

$$r = 0.802 \quad (8)$$

Dari perhitungan tersebut diketahui bahwa nilai r adalah 0.802. Selanjutnya jika dikaitkan dengan **Tabel 3. 4** tentang tingkatan atau interval reliabilitas, maka diketahui bahwa instrumen tersebut memiliki reliabilitas sangat tinggi dengan nilai $r = 0.802$.

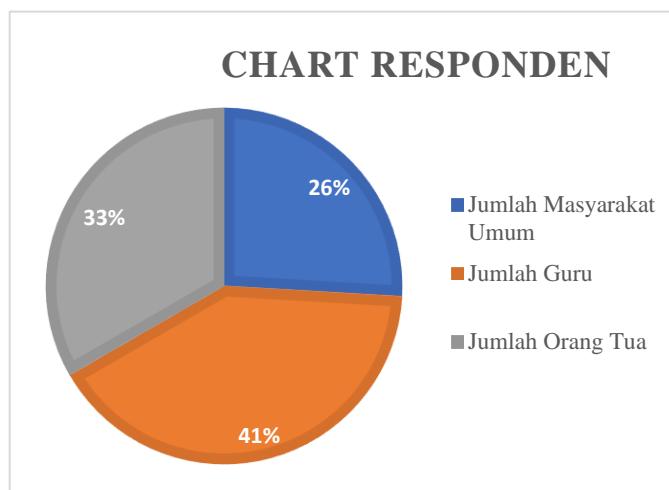
4.7.3. Uji Usability

Setelah melakukan uji validitas dan uji reliabilitas terhadap instrumen atau alat ukur pada kuesioner, maka selanjutnya kuesioner akan disebarluaskan secara terbuka. Adapun kuesioner akan berfokus pada guru-guru dari beberapa kelompok bermain, orang tua, dan masyarakat umum yang sebelumnya telah dibahas pada poin 3.9.1 terkait target kuesioner.

Dengan rentang waktu tujuh hari, berhasil mengumpulkan responden sebanyak 27 responden dengan masing-masing rincian dapat dilihat dari **Tabel 4. 13** atau dengan melihat diagram pada **Gambar 4. 38.**

Tabel 4. 13: Rincian responden

Jumlah Responden Guru	7
Jumlah Responden Orang Tua	11
Jumlah Responden Masyarakat Luar	9
Total	27



Gambar 4. 38: Chart rincian responden

Selanjutnya untuk hasil kuesioner dapat dilihat pada **Tabel 4. 14**. Adapun keterangan dari tabel tersebut adalah:

- a) R : Responden
- b) St : Status
- c) Q : Question
- d) T : Total
- e) G : Guru
- f) OT : Orang Tua
- g) MU : Masyarakat Umum

Tabel 4. 14: Tabel hasil kuesioner

R	St	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	T
R1	G	4	4	3	4	3	4	4	4	4	4	3	41
R2	OT	4	3	4	4	3	3	3	4	4	3	3	38
R3	G	4	4	4	3	4	4	4	4	3	4	4	42
R4	G	4	4	4	3	4	3	4	4	4	4	3	41
R5	G	4	3	4	3	4	4	4	4	4	4	3	41

R6	OT	4	4	4	4	4	4	4	4	4	4	4	4	44
R7	OT	4	4	4	3	4	4	4	4	4	4	3	4	42
R8	OT	4	3	4	4	4	4	4	4	4	4	3	4	42
R9	G	4	3	3	4	4	4	4	3	3	3	3	4	39
R10	OT	4	4	4	3	4	4	4	4	4	4	4	4	43
R11	G	4	4	4	4	3	4	3	4	4	4	4	3	41
R12	G	3	3	4	4	3	4	3	4	4	4	3	4	39
R13	G	4	4	3	3	4	4	4	4	4	4	3	3	40
R14	G	4	4	3	4	4	4	3	4	4	4	3	4	41
R15	OT	3	4	4	3	4	3	3	4	3	3	4	3	38
R16	OT	4	4	3	4	4	4	4	4	4	4	4	3	42
R17	OT	4	4	4	4	3	4	4	4	3	4	4	4	42
R18	OT	4	4	4	4	4	3	3	4	4	4	3	4	41
R19	MU	4	4	4	4	4	4	4	3	4	3	4	4	42
R20	MU	4	4	4	4	3	4	4	3	4	4	3	4	41
R21	MU	4	4	4	3	4	3	4	3	4	4	4	4	41
R22	MU	4	4	4	3	4	3	4	3	4	4	4	4	41
R23	G	4	4	3	3	4	3	3	4	4	3	4	4	39
R24	G	4	4	4	3	3	3	4	3	4	3	4	4	39
R25	MU	4	4	3	4	4	3	4	4	3	4	4	4	41
R26	MU	4	4	4	3	4	4	3	4	4	3	4	4	41
R27	MU	4	3	4	3	4	4	4	4	3	4	3	4	40

Berikutnya dengan menggunakan rumus pada persamaan 3, maka akan dihitung nilai *usability*. Sebagai contoh maka akan dilakukan perhitungan untuk melihat *usability* dari responden pertama.

$$usability = \frac{41}{44} \times 100 \quad (9)$$

$$usability = 93,2 \quad (10)$$

Dari perhitungan tersebut dapat dilihat bahwa *usability* atau kegunaan dari responden pertama mendapat nilai 93,2. Dari hasil ini, nilai kemudian akan dikonversikan ke tabel kategorinya sesuai pada **Tabel 3. 5**. Secara lebih lengkap, hasil dari perhitungan tersebut dapat dilihat pada **Tabel 4. 15** dengan keterangan:

- a) SP (%) : Skor Persentase
- b) SL : Sangat Layak
- c) L : Layak
- d) C : Cukup
- e) TL : Tidak Layak
- f) STL : Sangat Tidak Layak

Tabel 4. 15: Tabel hasil perhitungan usability

R	SKOR	SP (%)	K	R	SKOR	SP (%)	K
R1	41	93,18	SL	R15	38	86,36	SL
R2	38	86,36	SL		42	95,45	SL

R3	42	95,45	SL		R17	42	95,45	SL
R4	41	93,18	SL		R18	41	93,18	SL
R5	41	93,18	SL		R19	42	95,45	SL
R6	44	100	SL		R20	41	93,18	SL
R7	42	95,45	SL		R21	41	93,18	SL
R8	42	95,45	SL		R22	41	93,18	SL
R9	39	88,63	SL		R23	39	88,63	SL
R10	43	97,72	SL		R24	39	88,63	SL
R11	41	93,18	SL		R25	41	93,18	SL
R12	39	88,63	SL		R26	41	93,18	SL
R13	40	90,90	SL		R27	40	90,90	SL
R14	41	93,18	SL		Rata – Rata		92,76	SL

Dari hasil tabel di atas dapat dilihat bahwa dari 27 responden kuesioner tersebut memberikan rata-rata nilai 92,76% dengan total keseluruhan sebesar 1102 dari 1188 skor harapan. Dengan hasil ini dan hasil konversi nilai dengan intervalnya, menunjukkan bahwa hasil perhitungan *usability* pada *website* deteksi objek memiliki nilai “sangat layak”.

BAB V

PENUTUP

5.1. Kesimpulan

Berdasarkan rangkaian pelatihan, analisis, dan pengujian yang telah dilakukan pada bab sebelumnya, dapat ditarik beberapa kesimpulan bahwa, arsitektur YOLO (*You Only Look Once*) dapat digunakan untuk pengenalan objek dalam lingkungan pembelajaran anak-anak. Implementasi sistem pengenalan objek berbasis YOLO melalui platform *website* menggunakan Streamlit menunjukkan kinerja yang baik dalam mengenali dan mengidentifikasi berbagai objek di sekitar sesuai dengan objek yang telah ditentukan. Juga untuk pelatihan model pun diketahui bahwa dari ke-32 skenario atau percobaan yang dilakukan, percobaan ketujuh memiliki hasil mAP terbaik daripada percobaan-percobaan yang lainnya dengan nilai mAP sebesar 0,8791 atau sebesar 87%. Model yang dihasilkan ini pun mampu mendekripsi dengan baik yang ditunjukkan dengan hasil validasi model dan pengujian model yang dilakukan.

Pengujian instrumen pengukuran yang dilakukan baik itu pengujian validitas maupun pengujian reliabilitas memiliki hasil yang memuaskan di mana memiliki instrumen yang valid dengan tingkat kekonsistenan yang sangat tinggi. Dari hasil pengujian ini, selanjutnya pada pengujian dari aspek kegunaan atau *usability* yang dilakukan dengan menyebarkan kuesioner pun memiliki nilai “sangat layak” dengan rata-rata nilai sebesar 94,9%.

5.2. Saran

Dari penelitian yang telah dilakukan sebelumnya, terdapat beberapa saran yang dapat dilakukan untuk pengembangan selanjutnya. Adapun saran-saran

tersebut adalah untuk melakukan penelitian lebih lanjut untuk mencari sebab kesalahan prediksi pada model, khususnya pada objek sepatu. Selain itu, disarankan juga untuk memperbanyak variasi baik itu pada dataset maupun pada objek yang dideteksi sehingga model dapat mendeteksi objek yang lebih tepat dan lebih luas.

Pada website yang dihasilkan pun disarankan agar membuat website yang lebih interaktif sehingga mampu mengembangkan pola pikir anak dalam segi kognitifnya. Pengujian yang dilakukan pun disarankan untuk melihat aspek-aspek lainnya selain aspek kegunaan atau *usability* seperti aspek kesesuaian konten dan lainnya.

DAFTAR PUSTAKA

- Adarsh, P., & Rathi, P. (2020). YOLO v3-Tiny: Object Detection and Recognition using one stage improved model. *International Conference on Advanced Computing & Communication Systems*, 687–694.
- Adhinata, F. D., Wardhana, A. C., Rakhmadani, D. P., & Jayadi, A. (2020). Peningkatan Kualitas Citra pada Citra Digital Gelap. *Jurnal E-Komtek (Elektro-Komputer-Teknik)*, 4(2), 136–144. <https://doi.org/10.37339/e-komtek.v4i2.373>
- Aini, Q., Lutfiani, N., Kusumah, H., & Zahran, M. S. (2021). *Deteksi dan Pengenalan Objek dengan Model Machine Learning: Model YOLO*. 6(2), 2502–2714.
- Alberto, Joseph ; Hermanto, D. (2023). Klasifikasi Jenis Burung Menggunakan Metode CNN Dan Arsitektur ResNet-50. *Jurnal Teknik Informatika dan Sistem Informasi*, 10(3), 34–46.
- Andono, P. N., Sutojo, T., & Muljono. (2017). *Pengolahan Citra Digital* (A. Pramesta, Ed.; 1 ed.). ANDI.
- Ariansyah, D. S. (t.t.). Klasifikasi Hewan dengan Menggunakan Trasfer Learning Googlenet. *JIFT: Jurnal Informatika*.
- Ayuka, F., Pradana, P., Universitas, M., & Wacana, K. S. (2021). Pengembangan Instrumen Penilaian Sikap Disiplin Menggunakan Skala Likert Dalam Pembelajaran Tematik Kelas IV SD. *Jurnal Pendidikan Dasar*, 5(1), 13–29. <https://ejournal.stitpn.ac.id/index.php/fondatia>
- Banovbi, R., Irsal, P., & Utaminingrum, F. (2022). Sistem Pengenalan Gerak Kepala sebagai Navigasi Kursi Roda Pintar dengan menggunakan Metode

- YOLOV5 berbasis TX2. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, 6(12), 5576–5581. <http://j-ptiik.ub.ac.id>
- Budiyanta, N. E. (2018). Pengembangan Kelayakan Sistem Informasi Manajemen Untuk Workshop Dan Laboratorium. *JURNAL ELEKTRO*, 11(1), 1–14.
- Chandriah, K. K., & Naraganahalli, R. V. (2021). RNN / LSTM With Modified Adam Optimizer in Deep Learning Approach For Automobile Spare Parts Demand Forecasting. *Multimedia Tools and Applications*, 80(17), 26145–26159. <https://doi.org/10.1007/s11042-021-10913-0>
- Chen, H., Wang, Y., Guo, T., Xu, C., Deng, Y., Liu, Z., Ma, S., Xu, C., Xu, C., & Gao, W. (2021). Pre-Trained Image Processing Transformer. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12299–12310. <https://github.com>.
- Dhiyatmika, I. D. W., Putra, I. K. G. D., & Mandenni, N. M. I. M. (2015). Aplikasi augmented reality magic book pengenalan binatang untuk siswa TK. *Lontar Komputer*, 6(2), 120–127.
- Dwi Antoko, T., Azhar Ridani, M., & Eko Minarno, A. (2021). Klasifikasi Buah Zaitun Menggunakan Convolution Neural Network. *Komputika : Jurnal Sistem Komputer*, 10(2), 119–126. <https://doi.org/10.34010/komputika.v10i2.4475>
- Dwyer, B., & Gallagher, J. (2023, Maret 16). *Getting Started with Roboflow*. Roboflow Blog. <https://blog.roboflow.com/getting-started-with-roboflow/>
- Elshamy, R., Abu-Elnasr, O., Elhoseny, M., & Elmougy, S. (2023). Improving The Efficiency of RMSProp Optimizer by Utilizing Nestrove in Deep Learning. *Scientific Reports*, 13(1). <https://doi.org/10.1038/s41598-023-35663-x>

- Ernawati, I., & Sukardiyono, T. (2017). Uji Kelayakan Media Pembelajaran Interaktif Pada Mata Pelajaran Administrasi Server. *Elinvo: Electronics, Informatics, and Vocational Educational*, 2(2), 204–210.
- Fadillah, R. Z., Irawan, A., Susanty, M., & Artikel, I. (2021). Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO). *JURNAL INFORMATIKA*, 8(2).
- <http://ejournal.bsi.ac.id/ejurnal/index.php/ji>
- Gajalakshmi, P., Satyanarayana, J. V., Venkat Reddy, G., & Dhavale, S. (2020). Detection of Strategic Targets of Interest in Satellite Images using YOLO. *2020 4th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, 1–5.
- <https://doi.org/10.1109/ICCCSP49186.2020.9315197>
- Gibrani, H., Purnama, B., Kosala, G., & Pengoptimasian Pengukuran Kepadatan Jalan Raya, G. (t.t.). Optimizing Highway Density Measurement with CCTV Using the Yolov8 Method. *Technomedia Journal (TMJ)*, 9(1), 9.
- <https://doi.org/10.33050/tmj.v9i1.2216>
- Han, X., Zhang, Z., Ding, N., Gu, Y., Liu, X., Huo, Y., Qiu, J., Yao, Y., Zhang, A., Zhang, L., Han, W., Huang, M., Jin, Q., Lan, Y., Liu, Y., Liu, Z., Lu, Z., Qiu, X., Song, R., ... Zhu, J. (2021). Pre-Trained Models: Past, Present and Future. *AI Open*, 2, 225–250. <https://doi.org/10.1016/j.aiopen.2021.08.002>
- Hardi, N., & Sundari, J. (2022). Pengenalan Telapak Tangan Menggunakan Convolutionall Neural Network (CNN). *Jurnal Rekayasa Perangkat Lunak*, 4(1). <http://jurnal.bsi.ac.id/index.php/reputasi>

- Heri Pratikno, Muhammad Rifki Pratama, Yosefine Triwidayastuti, & Musayyanah. (2023). Pengenalan Gestur Jari Tangan Sebagai Media Pembelajaran Berhitung Bagi PAUD Berbasis Visi Komputer Dan Deep Learning. *Journal of Computer Electronic and Telecommunication*, 4(1). <https://doi.org/10.52435/complete.v4i1.355>
- Hidayatullah, P. (2017). *Pengolahan Citra Digital: Teori dan Aplikasi Nyata*. Informatika Bandung: Penerbit Informatika.
- Iryanto, S. Y., & Zaini, T. M. (2014). *Pengolahan Citra Digital*. Anggota IKAPI.
- Jiang, P., Ergu, D., Liu, F., Cai, Y., & Ma, B. (2022). A Review of Yolo Algorithm Developments. *Procedia Computer Science*, 199, 1066–1073.
- Jocher, G., & Sergiuwaxmann. (2023, Januari 9). *Ultralytics YOLOv8 Docs*. Ultralytics. <https://docs.ultralytics.com>
- Jönsson Hyberg, J., & Sjöberg, A. (2023). *Investigation Regarding The Performance of YOLOv8 in Pedestrian Detection* (TRITA-EECS-EX, Nomor 2023:282).
- Kaputa, D. S., & Landy, B. P. (2021). YOLBO: You only Look Back Once-A Low Latency Object Tracker Based on YOLO and Optical Flow. *IEEE Access*, 9, 82497–82507. <https://doi.org/10.1109/ACCESS.2021.3080136>
- Karlina, O. E., & Indarti, D. (2019). Pengenalan Objek Makaxnan Cepat Saji pada Video dan Real Time Webcam Menggunakan Metode Youu Only Look Once (YOLO). *Jurnal Ilmiah Informatika Komputer*, 24(3), 199–208. <https://doi.org/10.35760/ik.2019.v24i3.2362>
- Karna, N. B. A., Putra, M. A. P., Rachmawati, S. M., Abisado, M., & Sampedro, G. A. (2023). Toward Accurate Fused Deposition Modeling 3D Printer Fault

- Detection Using Improved YOLOv8 With Hyperparameter Optimization.
IEEE Access, 11, 74251–74262.
<https://doi.org/10.1109/ACCESS.2023.3293056>
- Khare, O. M., Gandhi, S., Rahalkar, A. M., & Mane, S. (2023). *YOLOv8-Based Visual Detection of Road Hazards: Potholes, Sewer Covers, and Manholes*.
<http://arxiv.org/abs/2311.00073>
- Komang Ayu, N., & Surya Manuaba, I. B. (2021). Media Pembelajaran Zoolfabeth Menggunakan Multimedia Interaktif untuk Perkembangan Kognitif Anak Usia Dini. *Jurnal Pendidikan Anak Usia Dini Undiksha*, 9(2), 194–201.
<https://ejournal.undiksha.ac.id/index.php/JJPAUD/index>
- Kulsum, U., & Cherid, A. (2023). Penerapan Convolutional Neural Network Pada Klasifikasi Tanaman Menggunakan ResNet50. *Simkom*, 8(2), 221–228.
<https://doi.org/10.51717/simkom.v8i2.191>
- Kumar, A., Kalia, A., Verma, K., Sharma, A., & Kaushal, M. (2021). Scaling Up Face Masks Detection with YOLO on a Novel Dataset. *Optik*, 239.
<https://doi.org/10.1016/j.ijleo.2021.166744>
- Kumari, N., Ruf, V., Mukhametov, S., Schmidt, A., Kuhn, J., & Küchemann, S. (2021). Mobile Eye-Tracking Data Analysis Using Object Detection via YOLO v4. *Sensors*, 21(22). <https://doi.org/10.3390/s21227668>
- Kusuma, W. A., Noviasari, V., & Marthasari, G. I. (2016). Analisis Usability dalam User Experience pada Sistem KRS-Online UMM menggunakan USE Questionnaire. *JNTETI: Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, 5(4), 294–301.

- Li, M., Ma, L., Blaschke, T., Cheng, L., & Tiede, D. (2016). A Systematic Comparison of Different Object-Based Classification Techniques Using High Spatial Resolution Imagery in Agricultural Environments. *International Journal of Applied Earth Observation and Geoinformation*, 49, 87–98.
<https://doi.org/10.1016/j.jag.2016.01.011>
- Lou, H., Duan, X., Guo, J., Liu, H., Gu, J., Bi, L., & Chen, H. (2023). DC-YOLOv8: Small-Size Object Detection Algorithm Based on Camera Sensor. *Electronics (Switzerland)*, 12(10). <https://doi.org/10.3390/electronics12102323>
- Maulana, A., Suherman, M., Masruriyah, A. F. N., & Novita, H. Y. (2024). Penerapan Algoritma CNN Menggunakan Framework YOLO Untuk Deteksi Objek Produk di Perusahaan Manufaktur. *INTI Nusa Mandiri*, 18(2), 107–114.
<https://doi.org/10.33480/inti.v18i2.5028>
- Maulana, F. (2021). *Machine Learning Object Detection Tanaman Obat Secara Real-Time Menggunakan Metode YOLO (You Only Look Once)*.
- Maurya, L. S., Hussain, M. S., & Singh, S. (2021). Developing Classifiers through Machine Learning Algorithms for Student Placement Prediction Based on Academic Performance. *Applied Artificial Intelligence*, 35(6), 403–420.
<https://doi.org/10.1080/08839514.2021.1901032>
- Maxwell, A. E., Warner, T. A., & Guillén, L. A. (2021). Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies—part 1: Literature review. *Remote Sensing*, 13(13).
<https://doi.org/10.3390/rs13132450>

- Mohammed, A. D., & Ekmekci, D. (2024). Breast Cancer Diagnosis Using YOLO-Based Multiscale Parallel CNN and Flattened Threshold Swish. *Applied Sciences (Switzerland)*, 14(7). <https://doi.org/10.3390/app14072680>
- Musdalifah, Anas, M., & Sadaruddin. (2020). Peningkatan Kreativitas Anak Melalui Metode Discovery pada Pembelajaran Sains di Taman Kanak-Kanak Aisyiyah Bustanul Athfal Mario. *TEMATIK: Jurnal Pemikiran dan Penelitian Pendidikan Anak Usia Dini*, 6(1), 42–52.
- Nalbant, K. G., & Uyanık, Ş. (2021). Computer Vision in the Metaverse. *Journal of Metaverse*, 1(1), 9–12.
- Nur, L., Hafina, A., & Rusmana, N. (2020). Kemampuan Kognitif Anak Usia Dini Dalam Pembelajaran Akuatik. *Scholaria: Jurnal Pendidikan dan Kebudayaan*, 10(1), 42–50.
- Panggalih, K., Kurniawan, W., & Gata, W. (2022). Implementasi Perbandingan Deteksi Tepi Pada Citra Digital Menggunakan Metode Roberst, Sobel, Prewitt dan Canny. *Infotek: Jurnal Informatika dan Teknologi*, 5(2), 337–347. <https://doi.org/10.29408/jit.v5i2.5923>
- Perez, F., Vasconcelos, C., Avila, S., & Valle, E. (2018). Data augmentation for skin lesion analysis. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11041 LNCS, 303–311. https://doi.org/10.1007/978-3-030-01201-4_33
- Piaget, J. (1952). The Origins of Intelligence in Children. Dalam M. Cook (Ed.), *The origins of intelligence in children*. W W Norton & Co. <https://doi.org/10.1037/11494-000>

- Priyono, F. H., Rahmawati, A., & Pudyaningtyas, A. R. (2021). Kemampuan Berpikir Simbolik Pada Anak Usia 5-6 Tahun. *Jurnal Kumara Cendekia*, 9(4), 212–217. <https://jurnal.uns.ac.id/kumara>
- Putra, Eka, W. S., & Soelaiman, R. (2016). Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) Pada Caltech 101. *Jurnal Teknik ITS*, 5(1), 76. <http://repository.its.ac.id/48842/>
- Putri, V. L., Wijayanti, A., & Kusumastuti, N. D. (2021). Pengembangan Media Frueelin Untuk Meningkatkan Perkembangan Kognitif Anak Usia Dini. *Jurnal Golden Age*, 5(02), 155–163. <https://doi.org/10.29408/jga.v5i01.3385>
- Rafly Alwanda, M., Putra, R., Ramadhan, K., & Alamsyah, D. (2020). Implementasi Metode Convolutional Neural Network Menggunakan Arsitektur LeNet-5 untuk Pengenalan Doodle. *Jurnal Algoritme*, 1(1), 45.
- 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*, 779–788. <http://arxiv.org/abs/1506.02640>
- Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint arXiv*. <http://arxiv.org/abs/1804.02767>
- Rosita, E., Hidayat, W., & Yuliani, W. (2021). Uji Validitas Dan Reliabilitas Kuesioner Perilaku Prososial. *FOKUS (Kajian Bimbingan & Konseling dalam Pendidikan)*, 4(4), 279. <https://doi.org/10.22460/fokus.v4i4.7413>
- Safita, M., & Suryana, D. (2022). Pengenalan Warna Melalui Media Stick Warna Terhadap Kemampuan Kognitif Anak Usia 4-5 Tahun. *Bunayya: Jurnal Pendidikan Anak*, 8(1), 28–43.

- Sandhya, & Kashyap, A. (2024). A Novel Method for Real-Time Object-Based Copy-Move Tampering Localization in Videos Using Fine-Tuned YOLO V8. *Forensic Science International: Digital Investigation*, 48. <https://doi.org/10.1016/j.fsidi.2023.301663>
- Sanjaya, J., & Ayub, M. (2020). Augmentasi Data Pengenalan Citra Mobil Menggunakan Pendekatan Random Crop, Rotate, dan Mixup. *Jurnal Teknik Informatika dan Sistem Informasi*, 6(2). <https://doi.org/10.28932/jutisi.v6i2.2688>
- Sari, D. H. A., Sa'idah, S., & Pratiwi, N. K. C. (2022). Klasifikasi Jenis Kulit Wajah Menggunakan Modifikasi Convolutional Neural Network (CNN). *e-Proceeding of Engineering*, 8(6), 3188–3194.
- Sholahuddin, M. R., Harika, M., Awaludin, I., Dewi, Y. C., Dhia Fauzan, F., Sudimulya, B. P., & Widarta, V. P. (2023). Optimizing YOLOv8 for Real-Time CCTV Surveillance: A Trade-off Between Speed and Accuracy. *Jurnal Online Informatika*, 8(2), 261–270. <https://doi.org/10.15575/join.v8i2.1196>
- Singh, S., Krishnan, S., & Research, G. (2020). Filter Response Normalization Layer: Eliminating Batch Dependence in the Training of Deep Neural Networks. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11237–11246.
- Suasapha, A. H. (2020). Skala Likert Untuk Penelitian Pariwisata; Beberapa Catatan Untuk Menyusunnya Dengan Baik. *JURNAL KEPARIWISATAAN*, 19(1), 26–37. <https://doi.org/10.52352/jpar.v19i1.407>
- Sufandi, U. U., Priono, M., Aprijani, D. A., Wicaksono, B. A., & Trihapnungsari, D. (2022). Uji Usabilityfungsi Aplikasi Web Sistem Informasi Dengan Use

- Questionnaire. (Studi Kasus: Aplikasi Web Sistem Informasi Tiras Dan Transaksi Bahan Ajar). *Jurnal Pendidikan Teknologi dan Kejuruan*, 19(1), 24–34.
- Sunarti, A., Yusuf Muslihin, H., & Abdul Muiz Lidinillah, D. (2023). Pengembangan Instrumen Deteksi Dini Perkembangan Kognitif Anak Usia 3 Tahun. *Jurnal PAUD Agapedia*, 7(1), 41–50.
<https://ejurnal.upi.edu/index.php/agapedia>
- Supriadi, M. F., Rachmawati, E., & Arifianto, A. (2021). Pembangunan Aplikasi Mobile Pengenalan Objek Untuk Pendidikan Anak Usia Dini. *Jurnal Teknologi Informasi dan Ilmu Komputer*, 8(2), 357–364.
<https://doi.org/10.25126/jtiik.2021824363>
- Terven, J. R., & Cordova-Esparza, D. M. (2023). A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, 5(4), 1680–1716.
<https://doi.org/10.3390/make5040083>
- Thoriq, M. Y. A., Permana, K. E., & Siradjuddin, I. A. (2023). Deteksi Wajah Manusia Berbasis One Stage Detector Menggunakan Metode You Only Look Once (YOLO). *JURNAL TEKNOINFO*, 17(1), 66–73.
<https://ejurnal.teknokrat.ac.id/index.php/teknoinfo/index>
- VYGOTSKY, L. S. (1978). *Development of Higher Psychological Processes* (M. Cole, V. Jolm-Steiner, S. Scribner, & E. Souberman, Ed.). Harvard University Press. <https://doi.org/10.2307/j.ctvjf9vz4>
- Wang, A. Y., Wang, D., Drozdal, J., Liu, X., Park, S., Oney, S., & Brooks, C. (2021). What Makes a Well-Documented Notebook? A Case Study of Data

Scientists' Documentation Practices in Kaggle. *Conference on Human Factors in Computing Systems - Proceedings.*

<https://doi.org/10.1145/3411763.3451617>

Wang, Q., Bi, S., Sun, M., Wang, Y., Wang, D., & Yang, S. (2018). Deep Learning Approach to Peripheral Leukocyte Recognition. *PLoS ONE*, 14(6). <https://doi.org/10.1371/journal.pone.0218808>

Wei, C., Kakade, S., & Ma, T. (2020). The Implicit and Explicit Regularization Effects of Dropout. Dalam *International conference on machine learning*. <https://github.com/>

Wen, L., Li, X., & Gao, L. (2021). A New Reinforcement Learning Based Learning Rate Scheduler for Convolutional Neural Network in Fault Classification. *IEEE Transactions on Industrial Electronics*, 68(12), 12890–12900. <https://doi.org/10.1109/TIE.2020.3044808>

Wu, X., Sahoo, D., & Hoi, S. C. H. (2020). Recent Advances in Deep Learning for Object Detection. *Neurocomputing*, 396, 39–64. <https://doi.org/10.1016/j.neucom.2020.01.085>

Yuni Wulandari, I., Indroasyoko, N., Mudia Alti, R., Asri, Y. N., & Hidayat, R. (2022). Pengenalan Sistem Deteksi Objek untuk Anak Usia Dini Menggunakan Pemrograman Python. *remik*, 6(4), 664–673. <https://doi.org/10.33395/remik.v6i4.11772>

Zhang, S., Wang, T., Wang, C., Wang, Y., Shan, G., & Snoussi, H. (2019). Video Object Detection base on RGB and Optical Flow Analysis. *2019 2nd China Symposium on Cognitive Computing and Hybrid Intelligence (CCHI)*, 280–284. [10.1109/CCHI.2019.8901921](https://doi.org/10.1109/CCHI.2019.8901921)

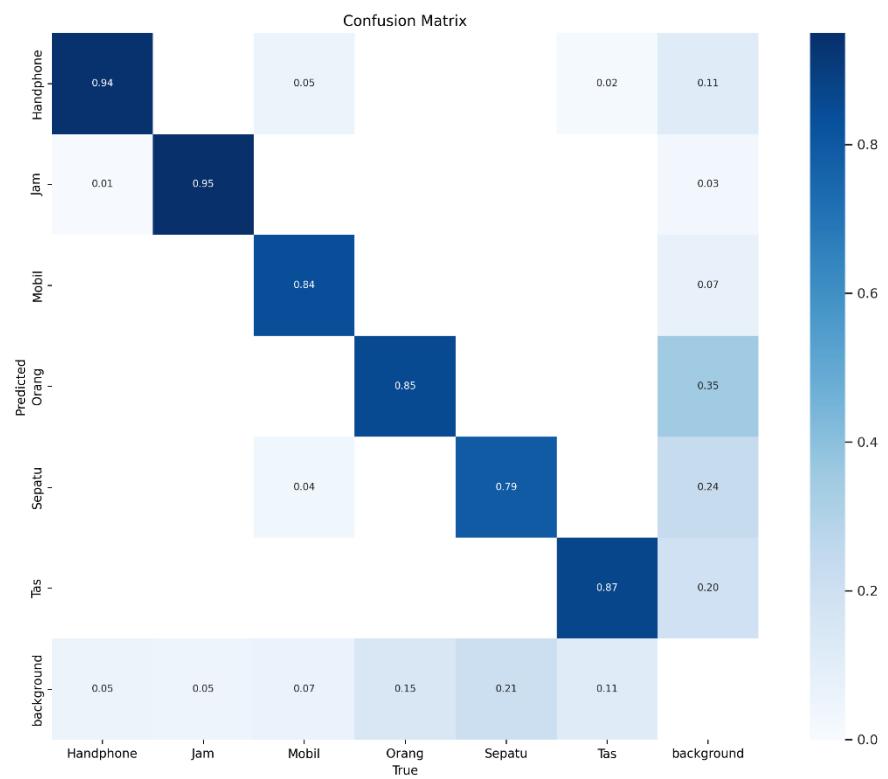
Zhao, L., & Li, S. (2020). Object Detection Algorithm Based on Improved YOLOv3. *Electronics* (Switzerland), 9(3).

<https://doi.org/10.3390/electronics9030537>

Zulwati, P. R., Fatmawati, F. A., & Agustina, R. (2022). Pengembangan Media Pembelajaran Pop Up Book Untuk Meningkatkan Perkembangan Kognitif Anak Usia 5-6 Tahun Di Tk Aba 42 GBA. *Jurnal Golden Age*, 6(02), 635–647. <https://doi.org/10.29408/goldenage.v6i02.77360>

LAMPIRAN

Lampiran 1: Gambar confusion matrix



Lampiran 2: Capture code dari function showDetectFrame

```
def showDetectFrame(conf, model, st_frame,
                    image, caption=None):

    res = model.predict(image, conf=conf)
    boxes = res[0].boxes
    labels = res[0].names
    res_plotted = res[0].plot()

    detected_labels = set()
    for box in boxes:
        label = labels[int(box.cls)]
        detected_labels.add(label)

    detected_labels = list(detected_labels)
    print("Detected labels:", detected_labels)

    st_frame.image(res_plotted, caption=caption,
                   channels="BGR", use_column_width=True)

def get_audio_bytes():
    text = (
        " ".join(detected_labels)
        if detected_labels
        else "Tidak ada objek yang terdeteksi"
    )
    print("Generated text for TTS:", text)

    tts = gTTS(text, lang="id")
    audio_buffer = io.BytesIO()
    tts.write_to_fp(audio_buffer)
    audio_buffer.seek(0)

    return audio_buffer

audio_buffer = get_audio_bytes()
audio_base64 = base64.b64encode(audio_buffer.read()).decode("utf-8")
audio_html = f"""
<audio autoplay>
    <source src="data:audio/mp3;base64,{audio_base64}" type="audio/mp3">
</audio>
"""
st.components.v1.html(audio_html, height=0)
```

Lampiran 3: Capture code dari function play_youtube

```
def play_youtube(conf, model):
    source_youtube = st.text_input("Silahkan Masukan Link YouTube")

    if st.button("Deteksi"):
        with st.spinner("Sedang Mendeteksi Objek ... "):
            try:
                yt = YouTube(source_youtube)
                stream = yt.streams.filter(file_extension="mp4", res=720).first()
                vid_cap = cv2.VideoCapture(stream.url)

                st_frame = st.empty()
                while vid_cap.isOpened():
                    success, image = vid_cap.read()
                    if success:
                        showDetectFrame(
                            conf, model, st_frame,
                            image, caption="Deteksi Video"
                        )
                    else:
                        vid_cap.release()
                        break
            except Exception as e:
                st.error("Ada Kesalahan Saat Memproses Link: " + str(e))
```

Lampiran 4: Capture code dari function live

```
def live(conf, model):
    webrtc_ctx = webrtc_streamer(
        key="object-detection",
        mode=WebRtcMode.SENDRECV,
        rtc_configuration={
            "iceServers": turn.get_ice_servers(),
            "iceTransportPolicy": "relay",
        },
        video_transformer_factory=lambda: VideoTransformer(model, conf),
        media_stream_constraints={"video": True, "audio": False},
        async_processing=True,
    )
```

Lampiran 5: Capture code dari class VideoTransformer

```
class VideoTransformer(VideoTransformerBase):
    def __init__(self, model, conf):
        self.model = model
        self.conf = conf
        self.last_detected_labels = set()

    def transform(self, frame):
        img = frame.to_ndarray(format="bgr24")
        res = self.model.predict(img, show=False, conf=self.conf)
        res_plotted = res[0].plot()

        detected_labels = set()
        for box in res[0].boxes:
            label = res[0].names[int(box.cls)]
            detected_labels.add(label)

        if detected_labels != self.last_detected_labels:
            self.last_detected_labels = detected_labels
            threading.Thread(target=self.speak_labels, args=(detected_labels,)).start()

        return res_plotted

    def speak_labels(self, detected_labels):
        text = (
            " ".join(detected_labels)
            if detected_labels
            else "Tidak ada objek yang terdeteksi"
        )

        tts = gTTS(text, lang="id")
        audio_buffer = io.BytesIO()
        tts.write_to_fp(audio_buffer)
        audio_buffer.seek(0) # Move the cursor to the start of the buffer

        pygame.mixer.init()
        pygame.mixer.music.load(audio_buffer, "mp3")
        pygame.mixer.music.play()
        while pygame.mixer.music.get_busy():
            continue
        pygame.mixer.quit()
```

Lampiran 6: Capture code dari function process_uploaded_video

```
def process_uploaded_video(conf, model):
    uploaded_video = st.file_uploader(
        "Silahkan Upload Video", type=["mp4", "avi", "mov"]
    )

    if uploaded_video is not None:
        with NamedTemporaryFile(delete=False, suffix=".mp4") as temp_file:
            temp_file.write(uploaded_video.read())
            temp_video_path = temp_file.name

        with open(temp_video_path, "rb") as video_file:
            video_bytes = video_file.read()
        if video_bytes:
            st.video(video_bytes)

    if st.button("Deteksi"):
        with st.spinner("Sedang Mendeteksi Objek ..."):
            try:
                vid_cap = cv2.VideoCapture(temp_video_path)
                st_frame = st.empty()
                while vid_cap.isOpened():
                    success, image = vid_cap.read()
                    if success:
                        showDetectFrame(
                            conf, model, st_frame, image, caption="Deteksi Video"
                        )
                    else:
                        vid_cap.release()
                        break
            except Exception as e:
                st.error("Error loading video: " + str(e))
```

Lampiran 7: Capture code dari function play_stored_video

```
def play_stored_video(conf, model):
    source_vid = st.selectbox(
        "Silahkan Pilih Video yang Sudah Disediakan", settings.VIDEOS_DICT.keys()
    )

    with open(settings.VIDEOS_DICT.get(source_vid), "rb") as video_file:
        video_bytes = video_file.read()
    if video_bytes:
        st.video(video_bytes)

    if st.button("Deteksi Video"):
        with st.spinner("Sedang Mendeteksi Objek ... "):
            try:
                vid_cap = cv2.VideoCapture(str(settings.VIDEOS_DICT.get(source_vid)))
                st_frame = st.empty()
                while vid_cap.isOpened():
                    success, image = vid_cap.read()
                    if success:
                        showDetectFrame(
                            conf, model, st_frame, image, caption="Deteksi Video"
                        )
                    else:
                        vid_cap.release()
                        break
            except Exception as e:
                st.error("Ada Kesalahan Saat Proses Video: " + str(e))
```

Lampiran 8: Capture code dari function take_picture

```
def take_picture(conf, model):
    picture = st.camera_input("Silahkan Mengambil Gambar")

    if picture:
        with NamedTemporaryFile(delete=False, suffix=".mp4") as temp_file:
            temp_file.write(picture.read())
            temp_pict_path = temp_file.name

        if st.button("Deteksi Foto"):
            with st.spinner("Sedang Mendeteksi Objek ... "):
                try:
                    vid_cap = cv2.VideoCapture(temp_pict_path)
                    st_frame = st.empty()
                    while vid_cap.isOpened():
                        success, image = vid_cap.read()
                        if success:
                            showDetectFrame(
                                conf, model, st_frame,
                                image, caption="Deteksi Gambar"
                            )
                        else:
                            vid_cap.release()
                            break
                except Exception as e:
                    st.error("Error loading video: " + str(e))
```

Lampiran 9: Capture code dari function up_picture

```
def up_picture(conf, model):
    source_img = st.file_uploader(
        "Silahkan Mengupload Gambar", type=("jpg", "jpeg", "png")
    )

    def proses():
        if source_img is not None:
            st_frame = st.empty()
            uploaded_image = PIL.Image.open(source_img)
            showDetectFrame(
                conf,
                model,
                st_frame,
                uploaded_image,
                caption="Hasil Deteksi Gambar",
            )

    col1, col2 = st.columns(2)
    with col1:
        try:
            if source_img is None:
                default_image_path = str(settings.DEFAULT_IMAGE)
                st.image(
                    default_image_path, caption="Gambar Awal",
                    use_column_width=True
                )
            else:
                st.image(source_img, caption="Gambar Awal",
                         use_column_width=True)
                if st.button("Deteksi", help="Klik tombol ini untuk deteksi"):
                    with st.spinner("Sedang Mendeteksi Objek ..."):
                        time.sleep(2)
                        with col2:
                            proses()

        except Exception as ex:
            st.error("Ada Kesalahan Saat Membaca File")
            st.error(ex)

    with col2:
        if source_img is None:
            default_image_path_result = str(settings.DEFAULT_DETECT_IMAGE)
            st.image(
                default_image_path_result,
                use_column_width=True,
                caption="Hasil Deteksi",
            )
```

Lampiran 10: Daftar pertanyaan dalam kuesioner

No	Pertanyaan	Sub Aspek
1.	Apakah Anda dapat dengan mudah memahami tujuan utama dari website ini?	Understandability
2.	Apakah navigasi atau arahan website ini mudah dipahami?	
3.	Apakah website ini menyediakan informasi yang cukup jelas?	
4.	Apakah website ini berfungsi secara efisien dalam pengenalan objek di lingkungan sekitar?	Operability
5.	Apakah Anda tidak mengalami kesulitan dalam menemukan fitur-fitur yang Anda butuhkan?	
6.	Bagaimana penilaian Anda terhadap desain keseluruhan dari website ini?	Attractiveness
7.	Apakah implementasi sistem antarmuka pada aplikasi dapat dipahami dengan mudah?	
8.	Apakah penggunaan warna, gambar, dan elemen desain lainnya menarik perhatian Anda?	
9.	Apakah Anda merasa mudah untuk belajar menggunakan website ini?	Learnability
10.	Apakah hasil deteksi yang dilakukan sesuai dengan objek yang dideteksi?	
11.	Apakah setelah menggunakan website ini, Anda merasa lebih familiar dengan tata cara penggunaannya?	