PERANCANGAN SISTEM PENDETEKSI PRODUK UNTUK MEMBANTU PEMAHAMAN PERALATAN TEMPAT TIDUR LANJUT USIA DENGAN PENERAPAN ARTIFICIAL INTELLINGENCE

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Strata-1
Pada Jurusan Teknik Industri Fakultas Teknologi Industri

Disusun Oleh:

Nama : Fahrul Triyulianto Rusli

No. Mahasiswa : 18522354

PROGRAM STUDI TEKNIK INDUSTRI

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2022

SURAT PERNYATAAN KEASLIAN TA

Demi Allah saya mengakui bahwa karya ini merupakan karya saya sendiri kecuali kutipan dan ringkasan yang salah satunya telah dicantumkan sumbernya. Jika ditemukan dikemudian hari ternyata terbukti pengakuan saya yang tidak benar dan melanggar peraturan yang sah dalam karya tulis ini dan hak kekayaan intelektual maka saya bersedia ijazah yang saya terima ditarik kembali oleh Universitas Islam Indonesia.

Yogyakarta, 10 Agustus 2022

Fahrul Triyulianto Rusli
18522354

SURAT KETERANGAN PELAKSANAAN TA

UNIVERSITAS ISLAM INDONESIA

FAKULTAS TEKNOLOGI INDUSTRI

Gedung KH. Mas Mansur
Kampus Terpadu Universitas Islam Indonesia
Jl. Kaliurang km 14,5 Yogyakarta 55584
T. (0274) 898444 ext 4110, 4100
F. (0274) 895007
E. fti@uii.ac.id
W. fti.uii.ac.id

SURAT KETERANGAN PENELITIAN

Nomor : 65/Ka.lab SIMANTI/20/Lab.SIMANTI/VIII/2022

Assalamu'alaikum Warohmatullahi Wabarokaatuh

Dengan hormat,

Yang bertanda tangan dibawah ini, menerangkan bahwa:

Nama	: Fahrul Triyulianto Rusli
Nim	: 18522354
Jurusan	: Teknik Industri
Dosen Pembimbing	: 1. Dr. Taufiq Immawan, S.T., MM.
	2. Abdullah 'Azzam, S.T., M.T

Menyatakan bahwa mahasiswa tersebut diatas telah melaksanakan penelitian tugas akhir dengan judul "**PERANCANGAN SISTEM PENDETEKSI PRODUK UNTUK MEMBANTU PEMAHAMAN PERALATAN TEMPAT TIDUR LANJUT USIA DENGAN PENERAPAN ARTIFICIAL INTELLINGENCE** "mulai pelaksanaan penelitian 18 Januari 2022 sampai 05 Agustus 2022.

Demikian surat keterangan penelitian ini kami buat. Atas perhatiannya dan kerja samanya kami mengucapkan terima kasih.

Wassalamu'alaikum Warohmatullahi Wabarokaatuh

Yogyakarta, 05 Agustus 2022

Kepala Laboratorium

Sistem Manufaktur Terintegrasi

Abdullah 'Azzam, S.T, M.T.

LEMBAR PENGESAHAN DOSEN PEMBIMBING

PERANCANGAN SISTEM PENDETEKSI PRODUK UNTUK MEMBANTU PEMAHAMAN PERALATAN TEMPAT TIDUR LANJUT USIA DENGAN PENERAPAN ARTIFICIAL INTELLINGENCE

TUGAS AKHIR

Disusun Oleh:

Nama : Fahrul Triyulianto Rusli

No. Mahasiswa : 18522354

Yogyakarta, 10 Agustus 2022

Dosen Pembimbing 1	Dosen Pembimbing 2
Dr. Taufiq Immawan. S.T., MM.	Abdullah 'Azzam, S.T., M.T

LEMBAR PENGESAHAN DOSEN PENGUJI

PERANCANGAN SISTEM PENDETEKSI PRODUK UNTUK MEMBANTU PEMAHAMAN PERALATAN TEMPAT TIDUR LANJUT USIA DENGAN PENERAPAN ARTIFICIAL INTELLINGENCE

TUGAS AKHIR

Disusun Oleh:

Nama : Fahrul Triyulianto Rusli

No. Mahasiswa : 18522354

Telah dipertahankan didepan siding penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata S-1 Teknik Industri

Tim Penguji

Dr. Taufiq Immawan, S.T., M.M
Ketua

Dr. Drs. Imam Djati Widodo, M.Eng.Sc.
Anggota I

Ir. M. Ridwan Andi Purnomo, S.T.,M.Sc., Ph.D., IPM
Anggota II

Abdullah 'Azzam, S.T., M.T.
Anggota III

Mengetahui,
Ketua Program Studi Teknik Indutsri
Fakultas Teknologi Industri
Universitas Islam Indonesia

Ir. M. Ridwan Andi Purnomo, S.T.,M.Sc., Ph.D., IPM

HALAMAN PERSEMBAHAN

Atas izin dan ridha Allah SWT saya persembahkan karya tulis ini kepada kedua orang tua saya Bapak Rusli H. Sirua dan Ibu Prapti yang tak pernah putus mendoakan saya sampai sekarang. Tak lupa juga saya persembahkan kara tulis ini kepada pembimbing saya Pak Azzam dan Pak Taufiq, serta sahabat-sahabat saya yang turut mendukung dan membantu saya dalam menyelesaikan karya tulis ini.

HALAMAN MOTTO

“Jika kamu tidak sanggup menahan lelahnya belajar, maka kamu harus menahan perihnya kebodohan”

-Imam Syafi’i

“Dan janganlah kamu (merasa) lemah, dan jangan (pula) bersedih hati, sebab kamu paling tinggi (derajatnya), jika kamu orang beriman.”

(QS. Ali ‘Imran: 140)

KATA PENGANTAR

Assalamu'alaikum warahmatullahi wabrakaatuh.

Alhamdulillahirabbil'alamiin segala puji bagi *Allah Subhanu Wa Ta'ala* atas segalah rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir dengan judul "Perancangan Sistem Pendeteksi Produk Untuk Membantu Pemahaman Peralatan Tempat Tidur Lanjut Usia Dengan Penerapan *Artificial Intellingence"* sebagai salah satu persyaratan untuk mendapat gelar Sarjana. Tidak lupa sholawat dan salam senantiasa penulis panjatkan kepada Nabi besar Muhammad Shallallahu 'Alaihi Wa Sallam beserta keluarga, sahabat, serta para pengikutnya hingga akhir zaman yang telah berjuang dan membimbing kita dari zaman kegelapan menuju jalan yang terang benderang untuk menggapai ridho Allah Subhanahu Wa Ta'ala.

Penyusunan Laporan Tugas Akhir ini kiranya tidak akan selesai tanpa ada bantuan dari berbagai pihak yang selalu membantu dan mendorong penulis untuk menyelesaikannya. Oleh karena itu, pada kesempatan kali ini penulis ingin mengucapkan terima kasih yang sebesari-besarnya kepada:

1. Bapak Prof. Ir. Hari Purnomo, M.T selaku Dekan Fakultas Teknologi Industri, Universitas Islam Indonesia.
2. Bapak Dr. Drs. Imam Djati Widodo, M.Eng.Sc., Selaku Ketua Jurusan Studi Teknik Industri, Fakultas Teknologi Industri, Universitas Islam Indonesia.
3. Bapak Ir. Muhammad Ridwan Andi Purnomo, S.T., M.Sc., Ph.D., Selaku Ketua Program Studi Teknik Industri, Fakultas Teknologi Industri, Universitas Islam Indonesia.
4. Bapak Abdullah 'Azzam, S.T., M.T dan Dr. Taufiq Immawan. S.T., MM. selaku dosen oembimbing Tugas Akhir yang telah membimbing dalam pelaksanaan dan penyusunan laporan Tugas Akhir.
5. Kedua orang tua penulis, Bapak Rusli H. Sirua dan Ibu Prapti yang telah memberikan doa, semangat, dan motivasi selama penyusunan hingga selesainya pengerjaan Tugas Akhir.
6. Kakak-Adik penulis serta keluarga yang selalu mendoakan dan memberikan semangat kepada penulis.
7. Sahabat seperjuangan "Anjay Bubar" Ari, Alya, Indika, Lukman, Sabil, dan Shandy yang menjadi *support system* selama pengerjaan Tugas Akhir.

8. Sahabat seperjuangan "Senapan Jogja" Ajeng, Luh Kadek, Safa, Safa, Citha, Farhah, Meydina, Naritha yang selalu mendukung dan mendoakan saya serta menjadi penghibur dikala penat selama proses pengerjaan Tugas Akhir.
9. Sahabat seperjuangan saya, Evanayeda Anindita, Rizqia Puteri Kinasih, Rizky Restiana, dan Rizki Rahmattullah yang senantiasa membantu saya dalam proses pengerjaan dan berbagi ilmu dan dukungan, serta mendengarkan keluh kesah penulis.
10. Keluarga besar Laboratorium Sistem Manufaktur Terintegrasi angkatan 2019 dan 2020, Laboran Pak Heri Susilo yang telah memberikan doa, semangat dan dukungan selama proses pengerjaan Tugas Akhir.

Demikian penulisan laporan Tugas Akhir ini disusun, penulis menyadari dalam penyusunan dan penulisan laporan Tugas Akhir ini masih memiliki kekurangan dan jauh dari sempurna. Oleh karena itu, penulis mengharapkan kritik, saran dan masukan yang bersifat membangun demi kesempurnaan penulisan di masa yang akan datang. Akhir kata semoga laporan Tugas Akhir ini dapat digunakan sebagai mana mestinya serta berguna khususnya bagi penulis dan bagi para pembaca pada umumnya.

Wassalamu'alaikum warahmatullahi wabrakaatuh.

Yogyakarta, 09 Agustus 2022

Fahrul Triyulianto Rusli

ABSTRAK

Lanjut Usia (Lansia) adalah seseorang yang mencapai usia 60 tahun ke atas. Indonesia adalah termasuk negara yang memasuki era penduduk berstruktur lanjut usia (*aging structured population*) karena jumlah penduduk yang berusia 60 tahun ke atas sekitar 7,18%. Meningkatnya populasi dari penduduk lansia dapat membawa begitu banyak dampak dalam kehidupan. Hal utama yang memiliki dampak besar pada meningkatnya lansia yaitu tingkat ketergantungan lansia. Ketergantungan yang sering di rasakan lansia disebabkan oleh kemunduran fisik maupun psikis. Perubahan fisik yang terlihat jelas pada lansia meliputi perubahan dari tingkat sel sampai ke semua sistem organ tubuh, salah satunya adalah penurunan penglihatan. beberapa lansia mengalami gangguan fungsi penglihatan akibat faktor usia, akan tetapi sebagian lain mengalami gangguan fungsi penglihatan dikarenakan penyakit pada mata yaitu katarak dan mata kering dan 60% aktivitas lansia digunakan diruang istirahat diantaranya mulai dari makan, membaca, tidur hingga berkomunikasi dengan yang lain. Dengan banyaknya aktivitas yang dilakukan oleh lansia diruang istirahat atau daerah tempat tidur dan gangguan atau penurunan penglihatan dapat menyebabkan peningkatan resiko yang berbahaya bagi lansia seperti jatuh, dan salah mengambil barang/alat. Untuk mengatasi permasalahan yang ada, dibutuhkan sebuah sistem dimana sistem tersebut dapat membantu lansia dalam melihat peralatan yang ada pada tempat tidur dengan penerapan *artificial intelligent* menggunakan metode *Convolutional Neural Network* (CNN). Berdasarkan metode CNN, didapatkan *model object detection* yang diuji menggunakan *images* dari setiap *label* atau *class*. Berdasarkan hasil pengujian didapatkan hasil tingkat akurasi yang sangat tinggi, yaitu sebesar 95% dengan hasil tersebut dapat diberikan rekomendasi sistem pada lansia yaitu dengan mengimplementasikan sistem yang sudah dibuat kepada suatu produk seperti kacamata untuk menciptakan sebuah inovasi *smartglasses* yang dapat mendeteksi objek dan ditransformasikan kedalam suara agar dapat didengarkan oleh *user* atau pengguna.

Keyword: *Artificial Intelligence, Convolutional Neural Network* (CNN)**,** *Deep Learning,* Lansia, *Object Detection*.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR NOTASI

Akurasi	:	Tingkat kedekatan hasil pengukuran terhadap nilai sebenarnya.
Batch Size	:	Jumlah sampel data yang disebarkan ke *neural network* atau ukuran dari satuan kecil *epoch* yang akan dimasukkan (*feeding*) kedalam komputer.
Class/Label	:	Variabel atau atribut yang digunakan dalam penelitian.
Checkpoint	:	Berkas atau *file* yang dihasilkan dari proses *training* yang disimpan dalam format .ckpt
Learning Rate	:	parameter *training* untuk menghitung nilai koreksi bobot pada waktu proses *training*
Loss	:	Ukuran seberapa jauh prediksi *model* tidak sesuai dengan label yang sebenarnya atau dengan kata lain ukuran seberapa buruk *model*
Epoch	:	*Dataset* telah melalui proses *training* pada *Neural Network* sampai kembali ke awal untuk satu kali putaran.
Step	:	Sejumlah langkah yang didefinisikan dalam konfigurasi *model* untuk proses *training* yang menentukan tingkat keberhasilan dari proses *training neural network*.
Model	:	Representasi dari apa yang telah dipelajari oleh sistem *Machine Learning* dari data *training*.
Object Detection	:	Proses memindai dan mencari atau mendeteksi objek dalam gambar atau video
Pipeline	:	Infrastruktur yang berkaitan dengan algoritma *Machine Learning* yang meliputi proses mengumpulkan data, memasukkan data kedalam *file* data pelatihan, melatih satu atau beberapa *model*, dan mengekspor *model*.
Central Processing Unit (CPU)	:	Perangkat keras komputer yang memahami dan menjalankan instruksi dari perangkat lunak.
Graphics Processing Unit (GPU)	:	*Processor* yang bertugas memproses instruksi oleh sebuah perangkat lunak dengan tujuan khusus yaitu pengolahan grafis.
Tensorboard	:	*Dashboard* yang menampilkan ringkasan yang disimpan selama satu atau beberapa program *TensorFlow*.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Lanjut usia (lansia) adalah seseorang dengan usia 65 tahun atau lebih yang terkadang menimbulkan masalah sosial, tetapi bukanlah suatu penyakit melainkan suatu proses natural tubuh meliputi terjadinya perubahan *deoxyribonucleic acid* (DNA), ketidaknormalan kromosom dan penurunan fungsi organ dalam tubuh. Sekitar 65% dari lansia yang mengalami gangguan kesehatan, hidup hanya ditemani oleh seseorang yang mengingatkan masalah kesehatannya, dan 35% hidup sendiri. Secara individu, pengaruh proses menua dapat menimbulkan berbagai macam masalah, baik masalah secara fisik, biologis, mental maupun masalah sosial ekonomi (Nies & McEwen, 2007).

Indonesia adalah termasuk negara yang memasuki era penduduk berstruktur lanjut usia (*aging structured population*) karena jumlah penduduk yang berusia 60 tahun ke atas sekitar 7,18%. Jumlah penduduk lansia di Indonesia pada tahun 2006 sebesar kurang lebih dari 19 juta, dengan usia harapan hidup 66,2 tahun. Pada tahun 2010 jumlah lansia sebanyak 18,04 Juta jiwa (7,18%) dan pada tahun 2010 mengalami peningkatan menjadi 23.992.553 jiwa (9,77%) sementara pada tahun 2011 jumlah lansia sebesar 20 juta jiwa (9,51%), dengan usia harapan hidup 67,4 tahun dan pada tahun 2020 diperkirakan sebesar 28,8 juta (11,34%), dengan usia harapan hidup 71,1 tahun (Kemenkes, 2012:2).

Berdasarkan data Badan Pusat Statistik pada tahun 2010 dan 2020 dengan judul Statistik Penduduk Lanjut Usia, didapatkan bahwasannya jumlah lanjut usia bertambah 2,33% dari tahun 2010 hingga 2020. Dimana pada tahun 2010 terdapat 7,59% atau setara dengan 18,04 juta jiwa, dan pada tahun 2020 mengalami kenaikan sebesar 2,33% menjadi 26 juta jiwa.

Meningkatnya populasi dari penduduk lansia dapat membawa begitu banyak dampak dalam kehidupan. Hal utama yang memiliki dampak besar pada meningkatnya lansia yaitu tingkat ketergantungan lansia. Ketergantungan yang sering di rasakan lansia disebabkan oleh kemunduran fisik maupun psikis, untuk tingkat kemandirian lansia akan terlihat saat melakukan aktivitas sehari-hari. Imobilitas fisik yang kurang juga mengakibatkan masalah yang diakibatkan dari berbagai masalah seperti fisik, psikologis dan lingkungan yang dirasakan lansia (Malida, 2011).

Perubahan fisik yang terlihat jelas pada lansia meliputi perubahan dari tingkat sel sampai ke semua sistem organ tubuh, diantaranya sistem pernapasan, pendengaran,

penglihatan, kardiovaskular, sistem pengaturan tubuh, *musculoskeletal*, *gastrointestinal*, *urogenital*, *endokrin* dan *integument*. Keseluruhan perubahan fisik diatas, ada salah satu yang semakin sering terjadi pada populasi yang menua adalah gangguan penglihatan. Timbulnya gangguan penglihatan di kemudian dapat mengubah kebiasaan hidup yang memiliki berbagai konsekuensi. Misalnya, orang tua dengan gangguan penglihatan lebih membatasi intensitas bergabung dengan rekan-rekan mereka, menyebabkan interaksi sosial berkurang (Renaud & Bédard, 2013).

Penurunan penglihatan merupakan keluhan yang besar bagi lanjut usia, sebab persepsi terhadap lingkungan berhubungan dengan rasa aman. Ketidakmampuan dalam menanggapi isyarat fungsi penglihatan inilah yang menyebabkan kesalahan dalam menangkap respon sensorik yang akan mengakibatkan kesulitan dalam memahami lingkungan geografis, bahaya, dan rangsang bergerak (Källstrand,2016).

Hasil studi pendahuluan menemukan beberapa lansia mengalami gangguan fungsi penglihatan akibat faktor usia, akan tetapi sebagian lain mengalami gangguan fungsi penglihatan dikarenakan penyakit pada mata yaitu katarak dan mata kering.

Berdasarkan penelitian yang dilakukan oleh Ummu Hindun Mastura (2016), menyatakan bahwa hampir 60% aktivitas lansia digunakan diruang istirahat diantaranya mulai dari makan, membaca, tidur hingga berkomunikasi dengan yang lain. Berdasarkan penelitian yang dilakukan oleh Roopa Rao pada tahun 2019 didapatkan bahwa untuk lansia, jatuh dapat mengakibatkan konsekuensi yang menghancurkan. Kebanyakan jatuh yang dialami oleh lansia terjadi di ruang yang diketahui dari pada didaerah yang tidak diketahui antara lain pada ruang tamu dan kamar tidur. Pada penelitian ini, didapatkan 86,7% nyaris celaka atau jatuh di daerah kamar tidur karena furnitur yang ada.

Dengan banyaknya aktivitas yang dilakukan oleh lansia diruang istirahat atau daerah tempat tidur dan gangguan atau penurunan penglihatan dapat menyebabkan peningkatan resiko yang berbahaya bagi lansia seperti jatuh, dan salah mengambil barang/alat (Ashar P. H, 2016). Untuk mengatasi terjadinya resiko yang berbahaya bagi lansia, dibutuhkan pemantauan perawatan secara lansgung. Perawatan kesehatan dari lansia berhubungan langsung dengan keseluruhan kebutuhan dari lansia dan pada hakikatnya perawatan lansia berhubungan erat dengan pemantauan secara langsung. Apabila lansia dilakukan perawatan secara *real-time* untuk mendapatkan data dan informasi yang aktual. Dan akan lebih baik jika informasi yang dihasilkan dari proses pemantauan merupakan prediksi akurat untuk beberapa waktu yang akan datang berdasarkan kesimpulan hasil pemantauan sebelumnya. Salah satu pemantauan yang dapat dilakukan adalah dengan pendekatan

teknologi. Teknologi menjadi salah satu faktor pendukung dalam pemantauan lansia. Alat-alat kesehatan, baik yang berada di klinik dan rumah sakit maupun yang banyak beredar di masyarakat, pada prinsipnya hal tersebut merupakan wujud dari penerapan teknologi.

Berdasarkan The Atma Jaya's Outlook for Development terkait "Kesejahteraan Warga Lanjut Usia Tantangan Kebijakan Kini dan Nanti" pada tahun 2022, salah satu teknologi yang dapat digunakan untuk membantu pemantauan pada lansia agar tidak terjadi peningkatan resiko adalah dengan penerapan kecerdasan buatan atau *Artificial Intelligence* (AI). Salah satu penerapan AI dalam penurunan resiko berbahaya yang terjadi oleh lansia adalah dengan teknik *computer vision* atau *object detection*, Penelitian aktivitas lansia berbasis AI dan *computer vision* saat ini merupakan salah satu bidang yang cukup diminati oleh peneliti. Dengan hanya menggunakan kamera dan data citra dari kamera, pemantauan aktivitas dapat dilakukan secara *real-time* dan kesimpulan berdasarkan komputasi di dalam AI dapat disajikan.

Terdapat berbagai metode yang dapat digunakan dalam *computer vision* (*object detection*) seperti *Support Vector Machine* (SVM), *K-Nearest Neighbors* (KNN) dan *Convolutional Neural Network* (CNN). Berdasarkan penelitian yang dilakukan oleh Mohammad Farid Naufal pada tahun 2019 mengenai "Analisis Perbandingan Algoritma SVM, KNN, dan CNN untuk Klasifikasi Citra Cuaca" didapatkan bahwa perbandingan untuk ketiga algoritma, CNN memiliki waktu eksekusi paling lama jika dibandingkan dengan KNN dan SVM, tetapi CNN mendapatkan akurasi yang paling baik dibandingkan dengan KNN dan SVM. Walaupun waktu eksekusi CNN untuk mendapatkan akurasi maksimal membutuhkan waktu rata-rata 458.49 detik, performa yang dihasilkan cukup signifikan jika dibandingkan dengan KNN dan SVM.

Berdasarkan penelitian yang dilakukan oleh Nur Jati Lantang Marfu'ah mengenai Perbandingan Antara Svm Dan Cnn Untuk Mendeteksi Objek Kapal Pada Citra Satelit pada tahun 2020, didapatkan bahwa SVM dan CNN memliki akurasi yang baik dimana SVM memiliki akurasi sebesar 94,38% sedangkan CNN sebesar 96,57% dan juga kemampuan mendeteksi objek oleh CNN lebih baik dibandingkan dengan SVM, walaupun CNN memiliki waktu yang lebih lama untuk mendeteksi objek dibanding SVM.

Berdasarkan permasalahan tersebut, peneliti akan melakukan perancangan suatu sistem dimana sistem tersebut dapat membantu lansia dalam melihat peralatan yang ada pada tempat tidur dengan penerapan *artificial intelligent* menggunakan metode *Convolutional Neural Network* (CNN).

1.2 Rumusan Permasalahan

Berdasarkan latar belakang diatas, maka rumusan masalah dalam penelitian ini adalah:

1. Bagaimana *model object detection* yang terbentuk untuk membantu pemahaman peralatan apa saja yang ada pada daerah tempat tidur?
2. Bagaimana tingkat akurasi yang dihasilkan untuk mendeteksi peralatan apa saja yang ada pada sekitar tempat tidur menggunakan *model* hasil *Convolutional Neural Network* (CNN)?
3. Bagaimana rancangan atau konsep penerapan *artificial intelligence* (AI) dalam bidang *computer vision* (*object detection*) untuk membantu lansia dalam pemahaman peralatan di daerah tempat tidur lanjut usia?

1.3 Batasan Permasalahan

Pada penelitian ini, terdapat beberapa batasan agar lebih spesifik dan terarah sesuai dengan apa yang dimaksudkan, maka diberikan batasan-batasan sebagai berikut:

1. Lanjut usia (lansia) yang ditargetkan merupakan lansia dengan rentang umur 62-80 yang memiliki penurunan fungsi pengelihatan.
2. *Software* yang digunakan adalah *Python* dengan *Framework Tensorflow.*
3. Data yang digunakan adalah data produk atau peralatan yang berada pada sekitar tempat tidur lanjut usia (Lansia).
4. Dataset gambar yang diambil melalui *crawling* dari *google image.*
5. Metode yang digunakan adalah *Convolutional Neural Network.*
6. Jumlah dataset yang digunakan berjumlah 100 gambar setiap class yang terdiri dari data *training* dan data *testing*.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah diatas, tujuan dalam penelitian ini adalah:

1. Mengetahui bagaimana penerapan *object detection* untuk membantu lansia dalam pemahaman peralatan di daerah tempat tidur.
2. Mengetahui *model* yang terbentuk dari hasil penelitian untuk melihat peralatan apa saja yang ada pada daerah tempat tidur lansia.
3. Mengetahui tingkat akurasi yang dihasilkan untuk mendeteksi peralatan apa saja yang ada pada tempat tidur lansia menggunakan *model* hasil *Convolutional Neural Network* (CNN).

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah sebagai berikut:

1. Proses dan hasil yang didapatkan dapat berguna sebagai langkah awal penerapan *Artificial Intelligence* (AI) sehingga dapat dikembangkan menjadi suatu produk yang dapat mendeteksi suatu produk atau peralatan lanjut usia agar dapat berguna sebagai alat bantu pemahaman atau penglihatan bagi lanjut usia (lansia).
2. Hasil penelitian ini dapat dijadikan acuan untuk penelitian lebih lanjut yang berbasis pada pendeteksian suatu objek dan terkait *Convolutional Neural Network* (CNN).

BAB II
TINJAUAN PUSTAKA

2.1 Kajian Literatur Induktif

Menurut Suriasumantri (2001), kajian induktif merupakan cara berpikir dimana ditarik suatu kesimpulan yang bersifat umum dari berbagai kasus yang bersifat individual. Kajian induktif didapatkan dari buku, jurnal, skripsi, tesis, artikel, situs internet, dan laporan penelitian terdahulu. Adapun berikut ini merupakan ringkasan dari penelitian terdahulu mengenai permasalahan pada penelitian

Tabel 2. 1 Kajian Induktif

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek						Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
1	Wildan Ahmad	Media bantu fungsi kognitif lansia berbasis citra digital dengan yolo	2022	100%		✓					✓					

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek						Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
2	Weiguo Feng, Rui Liu, Ming Zhu	Fall detection for elderly person care in a vision-based home surveillance environment using a monocular camera	2014	High Detection Accuracy			✓					✓				
3	Sekyoung Youm, Changgyun Kim, Seunghyun Choi & Yong-Shin Kang	Development of a methodology to predict and monitor emergency situations of the elderly based on object detection	2018	90%				✓					✓			

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek						Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
4	Viet Dung Nguyen, Minh Thao Le, Anh Duc Do, Hoang Hai Duong, Toan Dat Thai, Duc Hoa Tran	An Efficient Camera-based Surveillance for Fall Detection of Elderly People	2014	92.5%			✓							✓		
5	Kun-Lin Lu and Edward T.-H. Chu	An Image-Based Fall Detection System for the Elderly	2018	95,96%			✓								✓	
6	Sudipta Chatterjee, Sahadev Roy	A low-cost assistive wheelchair for handicapped & elderly people	2021	-							✓					

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek							Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
7	Pitchaya Chotivatunyu, Narit Hnoohom	Medicine Identification System on Mobile Device for The Elderly Deep Learning based	2020	93,79%					✓							✓
8	Bing Lyu, Zhichen Wang, Hengyi Li, etc	Medicine Packaging Information Recognition for Medication use in the Elderly	2020	97,16%					✓							✓
9	Leena K. Gautam & VijayS. Gulhane	Food Assessment *Model* for Indian Elderly Persons Using CNN	2022	97%						✓						✓

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek						Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
		and Image Processing Techniques														
10	Pisol Ruenin, Jakramate Bootkrajang, Jakarin Chawachat'	A System to Estimate the Amount and Calories of Food that Elderly People in the Hospital Consume	2020	-						✓			✓			✓

No	Penulis	Judul	Tahun	Tingkat Akurasi	Objek						Metode					
					Peralatan Kamar Tidur	Peralatan Ruang Tamu (Remote AC, TV, HP, dll)	Human (Manusia yang Jatuh)	Pose Manusia	Obat-Obatan	Kalori dan Nutrisi	YOLO	SVM	Fast-RCNN	Rule-Based	IFADS	CNN
11	Fahrul Triyulianto Rusli	Perancangan Sistem Pendeteksi Produk Untuk Membantu Pemahaman Peralatan Tempat Tidur Lanjut Usia Dengan Penerapan Artificial Intellingence	2022	95%	✓											✓

Berdasarkan tabel diatas, terdapat penelitian terdahulu terkait *object detection* dengan lansia antara lain Peralatan Ruang Tamu, Human (Manusia yang Jatuh), Pose Manusia, Obat-Obatan dan Kalori dan Nutrisi, dengan menggunakan berbagai metode. Berdasarkan latar belakang, kebanyakan jatuh yang dialami oleh lansia terjadi di ruang yang diketahui dari pada didaerah yang tidak diketahui antara lain pada ruang tamu dan kamar tidur, pada penelitian sebelumnya sudah terdapat penelitian terkait *object detection* pada produk di ruang tamu, dan pada penelitian yang dilakukan oleh Riopa Rao didapatkan 86,7% nyaris celaka atau jatuh di daerah kamar tidur, sehingga penelitian terkait *object detection* pada kamar tidur lansia belum ada dan menjadi penting untuk diteliti.

2.2 Kajian Literatur Deduktif

2.2.1 Lanjut Usia

Lanjut Usia (Lansia) merupakan seseorang yang telah mencapai usia 60 tahun atau lebih. Secara global populasi lansia telah diprediksi akan terjadi peningkatan seiring berjalannya waktu. Setelah tahun 2100, Indonesia diprediksi akan mengalami peningkatan populasi lansia dan lebih tinggi jika dibandingkan dengan populasi lansia di dunia (Infodatin, 2016).

2.2.2 *Object Detection*

Deteksi objek (*Object Detection*) merupakan salah satu teknik dari bidang *computer vision* untuk mendeteksi suatu objek dalam gambar atau video. Machine learning merupakan algoritma yang biasa digunakan untuk menghasilkan hasil yang bermakna. Tujuan dari *object detection* adalah melakukan tiruan kecerdasan yang dimiliki oleh manusia dalam melihan suatu objek dengan bantuan *computer* (Sahasri M & Gireesh. C, 2017).

2.2.3 *Artificial Intelligence* (AI)

Artificial Intelligence (Kecerdasan Buatan) merupakan salah satu bidang pengetahuan dalam ilmu komputer yang biasa dikenal dengan AI. AI atau *Artificial Intelligence* merupakan suatu ilmu yang membahas bagaimana membangun suatu sistem komputer yang menunjukkan kecerdasan dalam berbagai cara. Masalah inti dari AI meliputi pemrograman komputer untuk sifat-sifat tertentu seperti pengetahuan, pemikiran, penyelesaian masalah, persepsi, pembelajaran, perencanaan, dan kemampuan dalam memanipulasi serta memindahkan objek. AI merupakan salah satu area penelitian yang dinamis dalam topik riset ilmu komputer. Sampai saat ini, telah banyak dilakukan suatu penelitian terkait perkembangan AI diantaranya *neural network*, *evolutionary computing*, *machine learning*, *natural language processing*, pemrograman otomatis, robotika, dan *object-oriented programming* (Yunanto Andhik Ampuh, 2016).

2.2.4 *Deep Learning*

Secara definisi *deep learning* merupakan bagian dari *machine learning* yang digunakan untuk pemodelan abstraksi tingkat tinggi pada suatu data berdasarkan algoritma dengan menggunakan lapisan implementasi dan menggunakan struktur yang kompleks atau sebaliknya, terdiri dari beberapa transformasi non-linear (Fikrie Abdillah, 2016).

Deep learning memiliki salah satu fitur yaitu, *Feature Engineering*. *Feature Engineering* digunakan untuk melakukan ekstraksi pola yang terbentuk atau didapatkan dari data yang akan membantu model untuk mempelajar karakteristik dari data sehingga

model dapat membedakan kelas, sehingga fitur tersebut juga berperan untuk mendapatkan hasil prediksi yang maksimal (Santoso & Ariyanto, 2018).

Perbedaan antara *machine learning* dengan *deep learning* dapat dilihat dari cara proses pembelajaran yang dilakukan, *machine learning* menggunakan pembelajaran representasi tingkat tunggal sedangkan *deep learning* menggunakan gabungan dari beberapa lapisan pemrosesan untuk proses pembelajaran, dengan demikian membuat *deep learning* mampu menyelesaikan tugas komplek pada suatu data yang memiliki dimensi tinggi karena pembelajaran representasi yang tinggi (Goodfellow, 2016). Contoh dalam penerapan *deep learning* adalah dengan kemampuan komputasi dan memori yang lebih besar, seseorang dapat membuat *deep neural network* atau suatu jaringan saraf dari banyaknya lapisan, dengan keberhasilan dalam mempelajari *deep neural network*, *deep learning* saat ini berkembang menjadi lebih canggih untuk melakukan suatu deteksi, klasifikasi, segmentasi dan *recognition* pada proses identifikasi dan verifikasi suatu obek dalam gambar.

2.2.5 *Computer Vision Technology*

Vision secara bahasa memiliki arti yaitu penglihatan, dan juga dapat diartikan sebagai suatu proses pengamatan melalui panca indra penglihatan manusia. *Computer vision* merupakan suatu pembelajaran untuk menganalisis suatu gambar dan video untuk mendapatkan hasil seperti yang bisa dilakukan manusia, oleh karena itu *computer vision* mencoba untuk meniru cara kerja sistem pengelihatan yang dilakukan oleh manusia (*Human Vision*).

Secara garis besar, *computer vision* merupakan suatu teknologi yang digunakan oleh sebuah komputer agar dapat melihat seperti layaknya manusia, kemampuan tersebut utnuk mengenali suatu objek merupakan kombinasi dari pengolahan citra (*image processing)* dan pengenalan pola (*pattern recognition)*. *Output* dari *computer vision* adalah suatu deskripsi atau interpretasi dalam adegan 3D (Le, 2015).

2.2.6 *Artificial Neural Network*

Neural Network adalah prosesor yang terdistribusi paralel, terbuat dari unit-unit yang sederhana, dan memiliki kemampuan untuk menyimpan pengetahuan yang diperoleh secara eksperimental dan siap pakai untuk berbagai tujuan (Nurmila, Sugiharto, & Sarwoko, 2010).

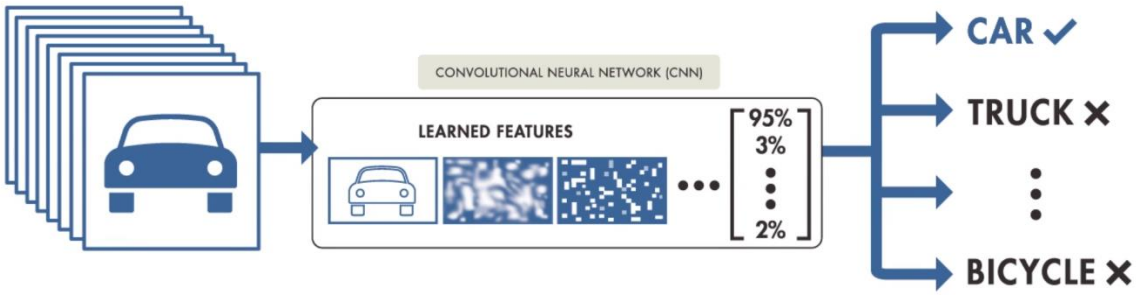
Artificial Neural Network (ANN) merupakan sebuah sistem pembelajaran terhadap penerimaan informasi yang memiliki kinerja layaknya sebuah jaringan syaraf pada

manusia. ANN diimplementasikan dengan menggunakan program komputer sehingga mampu menyelesaikan sejumlah proses perhitungan.

ANN adalah suatu metode pengelompokan dan pemisahan data yang prinsip kerjanya sama seperti *neural network* pada manusia. Elemen mendasar dari paradigma tersebut adalah struktur yang baru dari sistem pemrosesan informasi. ANN dibentuk untuk memecahkan suatu masalah tertentu seperti pengenalan pola atau klasifikasi dari proses pembelajaran (Muslimin. M, 2016).

2.2.7 *Convolutional Neural Network*

Convolutional network atau yang dikenal juga dengan nama *Convolutional Neural Network* (CNN) adalah tipe khusus dari neural network untuk memproses data yang mempunyai topologi jala atau *grid-like topology*. Pemberian nama *convolutional neural network* mengindikasikan bahwa jaringan tersebut menggunakan operasi matematika yang disebut konvolusi. Konvolusi sendiri adalah sebuah operasi linear. Jadi *Convolutional Network* adalah *neural network* yang menggunakan konvolusi minimal satu pada lapisannya (Lecun, Bengio, & Geoffrey, 2015). CNN digunakan untuk melakukan klasifikasi data yang berlabel dengan menggunakan metode *supervised learning* yang cara kerjanya adalah terdapat data yang dilatih dan terdapat variabel yang ditargetkan sehingga tujuan dari metode ini yaitu mengelompokkan suatu data ke data yang sudah ada.



Gambar 2. 1 Ilustrasi Arsitektur *Convolutional Neural Network*

CNN akan belajar memprediksikan label gambar yang diberikan sesuai dengan representasi *feature* mulai dari yang sederhana hingga yang lebih kompleks. *Feature* ini yang kemudian digunakan dalam jaringan untuk memprediksi kategori dengan benar.

Convolutional Neural Network merupakan metode yang tercakup di dalam kelas *Feed Forward Neural Network* yang terinspirasi dari *visual cortex* dari otak dan dikhususkan untuk memproses data yang memiliki struktur *grid*. CNN mempunyai beberapa jenis layer yang digunakan.

Arsitektur CNN terdiri dari 2 bagian utama yaitu *Feature Extraction Layer* dan *Fully-Connected Layer*.

1. *Feature Extraction Layer*

Proses yang terjadi pada bagian ini adalah "encoding" dari sebuah image menjadi features yang berupa angka-angka yang merepresentasikan image tersebut (Feature Extraction). Feature extraction layer terdiri dari dua bagian. Adapun layer pertama yaitu Convolutional Layer dan layer kedua merupakan pooling layer. Pada setiap layer terdapat fungsi aktivasi. Layer ini menerima input gambar secara langsung dan memprosesnya sehingga menghasilkan output berupa vektor untuk diolah pada layer berikutnya.

2. *Fully-Connected Layer*

Tersusun dari beberapa layer dan pada setiap layer tersusun atas neuron yang terkoneksi secara penuh (fully connected) dengan layer lainnya. Layer ini menerima input dari hasil output Feature Extraction Layer (layer ekstrasi fitur) gambar berupa vektor dan kemudian ditransformasikan dengan tambahan beberapa hidden layer. Hasil output yang dihasilkan adalah berupa skoring kelas untuk klasifikasi (Zufar & Setiyono, 2016).

BAB III
METODE PENELITIAN

3.1. Subjek Penelitian

Subjek dari penelitian yang dilakukan adalah peralatan yang ada pada tempat tidur lanjut usia.

3.2. Pengumpulan Data

3.2.1 Jenis Data

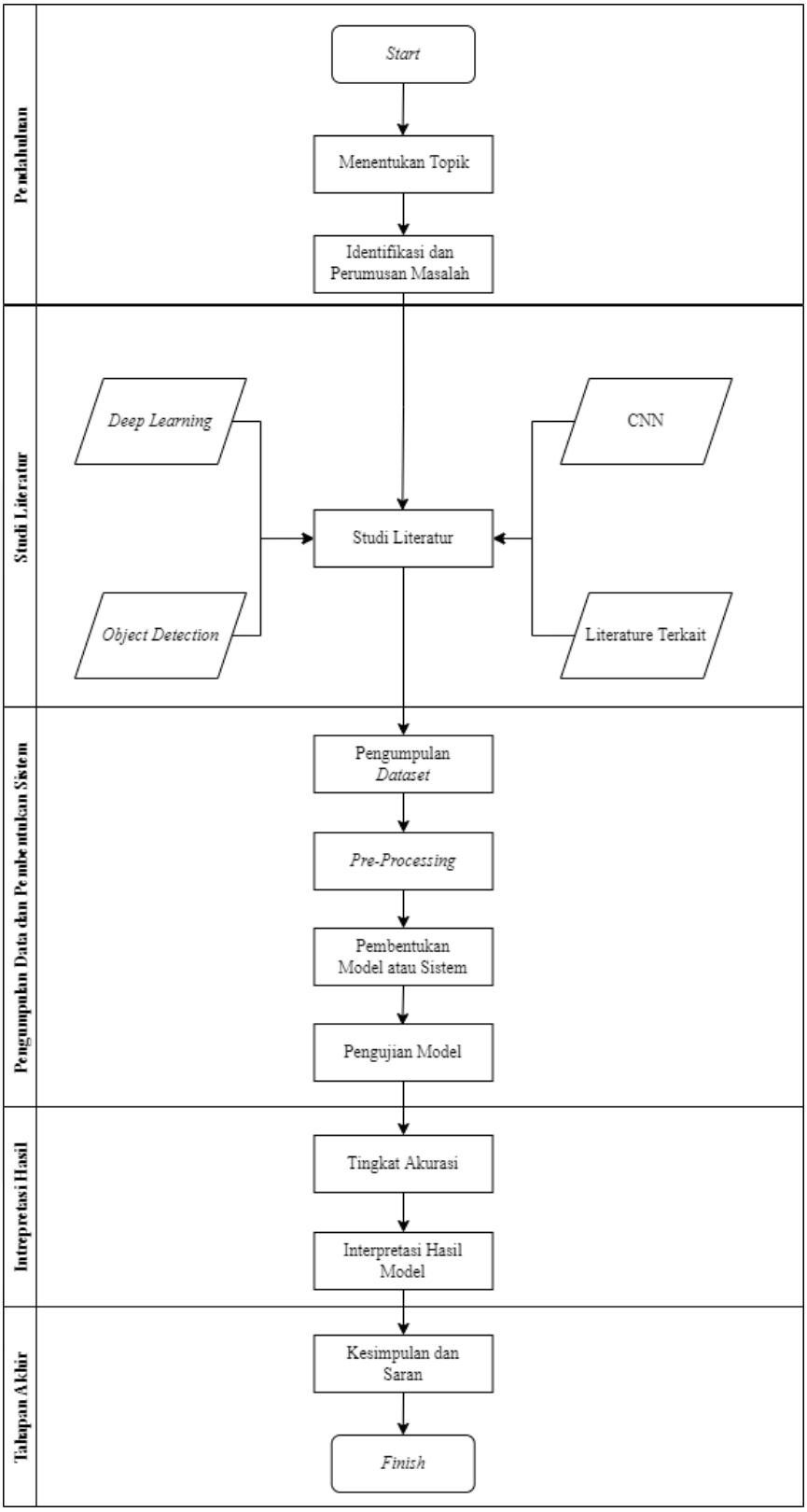
Pada penelitian ini, jenis data yang digunakan adalah data primer dan data sekunder. Data primer yang digunakan adalah data jenis barang atau peralatan yang digunakan lanjut usia pada tempat tidur, dimana jumlah tempat tidur lansia yang digunakan sebanya 10 kamar. Sedangkan data sekunder yang digunakan adalah *dataset* berupa gambar produk yang diambil melalui *crawling* dari *google image* dengan bantuan *tools* atau *extension Google Chrome* yaitu *Fatkun Batch Download Image* pada *Google Images*.

3.2.2 Metode Pengumpulan Data

Pada penelitian ini, metode untuk melakukan pengumpulan data didapatkan dari melakukan observasi dan studi literature. Observasi yang dilakukan adalah dengan melihat langsung kamar tidur lansia untuk memperoleh informasi yang dibutuhkan, dimana hasil observasi ini akan digunakan sebagai input dalam pembuatan dataset. Selain melakukan observasi langsung, peneliti melakukan pengumpulan data menggunakan studi literatur berupa data *images* atau gambar dari setiap produk pada kamar tidur lansia yang digunakan dengan menggunakan teknik *crawling* dari *google images* dengan bantuan *tools* atau *extension Google Chrome* yaitu *Fatkun Batch Download Image.*

3.3. Alur Penelitian

Berikut merupakan alur penelitian ini:



Gambar 3. 1 Alur Penelitian

Pada penelitian ini adapun tahap-tahap yang dilakukan sebagai berikut:

1. Langkah pertama yang dilakukan dalam penelitian ini adalah menentukan topik yang akan dilakukan pada penelitian ini

2. Langkah selanjutnya peneliti mengidentifikasi dan merumuskan masalah yang dapat diteliti dalam penelitian ini
3. Selanjutnya peneliti melakukan studi literatur yang berkaitan dengan topik yang telah ditentukan, seperti *Deep Learning, Object Detection, Convolutional Neural Network* (CNN), dan literature terkait (Lanjut Usia, *Artificial Intelligence, Computer Vision Technology, Artificial Neural Network, Python,* dan *Tensorflow*).
4. Selanjutnya merupakan tahapan pengumpulan data dan pe*model*an sistem. Dimulai dari pengumpulan *dataset,* dimana data *image* yang digunakan ada data produk yang ada didaerah sekitar tempat yang didapatkan melalui *crawling* dari *google image*.
5. Langkah selanjutnya adalah tahap *pre-processing,* dimana tahapan tersebut terdapat beberapa proses diantara lain:
    a. Tahapan pertama pada *pre-processing* adalah Pelabelan Data, dimana *dataset* yang telah dikumpulkan akan diberikan label satu persatu sesuai dengan *class* objek tersebut dengan bantuan *library LabelImg*. Pelabelan gambar ini bertujuan untuk menyimpan seluruh informasi gambar yang selanjutnya disimpan dalam *file* ".XML" dengan format PASCAL VOC.
    b. Selanjutnya hasil *output* berupa *file* dengan format .XML (*Xtensible Markup Language*) yang akan dikonversi menjadi .csv agar dapat terbaca pada *framework Tensorflow*.
    c. Selanjutnya *dataset* dalam format .csv akan dikonversikan menjadi format *TFRecord* and *Tensorflow Record* yang merupakan format penyimpanan dari *Tensorflow*. *File TFRecord* tersebut akan digunakan dalam proses *feeding data* atau membaca data *input* sehingga informasi *dataset* dapat diambil secara langsung dengan fungsi *feed_dictionary*.
    d. Langkah terakhir pada tahapan *pre-processing* adalah proses pembuatan *label map,* dimana label map dilakukan untuk mendefenisikan setiap class objek dalam kalkulasi. Pada penelitian ini menggunakan 25 objek, sehingga label map yang dibuat berjumlah 25 item yang terdiri dari ID dan Nama yang harus sesuai dengan urutan dan nama saat proses pelabelan data.
6. Setelah tahapan *pre-processing* data, peneliti akan melakukan pembentukan *model* atau sistem dari *object detection*, dimana pada tahapan pembentukan *model* atau sistem ini dilakukan dengan membuat *konfigurasi pipeline* menggunakan *ProtoBuf* yang akan digunakan untuk mengkonfigurasi proses *training* dan proses *testing*.

Pada proses *training model TFRecord* akan digunakan sebagai *input* data dalam pelatihan *model*. Metode *convolutional neural network* (CNN) akan berperan pada proses *training model*, dimulai dari tahapan *Convolution + ReLu*, dimana pada tahapan ini *dataset* akan dilakukan operasi konvolusi dari layer sebelumnya dan akan dilanjutkan pada proses fungsi aktivasi *Rectifier Linier Unit* (ReLu) yang memiliki fungsi untuk mengubah nilai negative (-) menjadi positif (+) dari hasil konvolusi yang didapatkan. Selanjutnya hasil tahapan *Convolution + ReLu*, akan memasuki tahapan *pooling* untuk pengurangan dimensi dari *feature map* dengan menggunakan *max pooling*. Tahapan selanjutnya adalah *flattening*, dimana *feature map* dari hasil *pooling* akan diubah menjadi bentuk *vector*, dimana *vector* ini akan menjadi input untuk memasuki tahapan akhir dari metode CNN yaitu tahapan *fully connected layer*. Setelah proses *training* selesai, tahapan selanjutnya adalah *model* yang dihasilkan dari proses *training* akan dilakukan *export graph* untuk mengekspor *model* yang akan digunakan untuk menjadi pendeteksi objek.

7. Setelah melakukan *export graph* peneliti akan melakukan uji coba dari *model* atau sistem yang telah didapatkan untuk mendeteksi suatu objek, apakah sistem tersebut sudah sesuai atau belum dan akan mendapatkan hasil berupa tingkat akurasi hasil deteksi.
8. Setelah mendapatkan hasil dari uji coba *model*, langkah selanjutnya melakukan interpretasi pada hasil pendeteksian yang kemudian akan didapatkan kesimpulan dan saran pada penelitian ini.

BAB IV
PEMBANGUNAN SISTEM

4.1 Pengumpulan Data

Pengumpulan data didapatkan dari melakukan observasi dan studi literature. Observasi yang dilakukan adalah dengan melihat langsung kamar tidur lansia untuk memperoleh informasi yang dibutuhkan, dimana pada penelitian ini hanya menggunakan 10 data kamar lansia, dikarenakan terdapat kesulitan untuk mencari lansia yang bersedia untuk kamarnya dilakukan observasi. Untuk memperbanyak jumlah data yang digunakan, peneliti melakukan penambahan 100 data *images* setiap produk dari *google images* dengan teknik *crawling*, dimana penambahan data ini berguna untuk membantu proses *training* pada *model* yang akan dibentuk.

4.2.1 Data Kamar Lansia

Berdasarkan hasil pengumpulan data, didapatkan 10 kamar lansia sebagai berikut:

1. Kamar 1

   Biodata Lansia kamar 1:

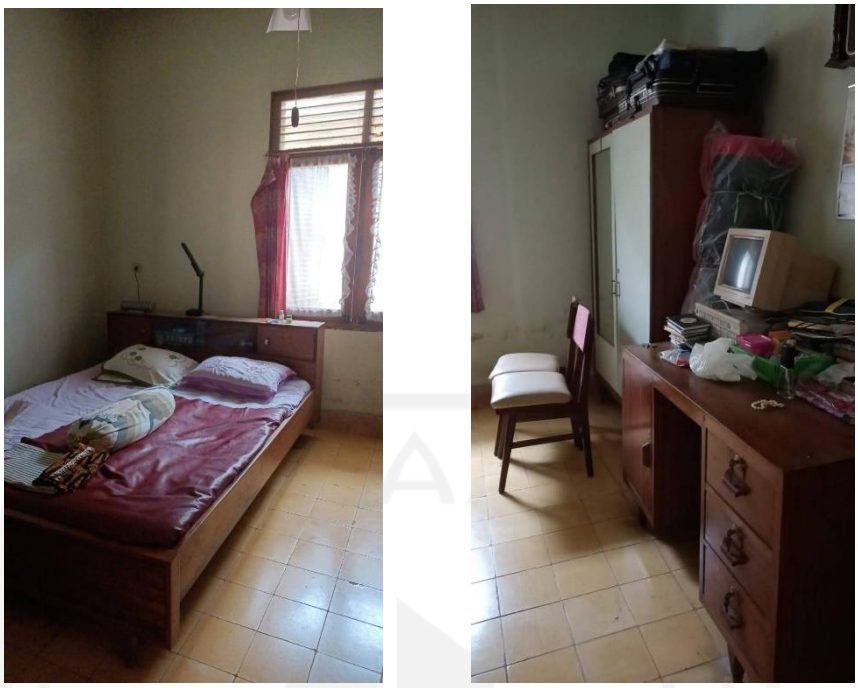
   Nama : Aan Sumarni

   TTL : Tasikmalaya, 23 Febuari 1942

   Umur : 80 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 1:



Gambar 4. 1 Gambar Kamar 1

2. Kamar 2

   Panti Jompo Madania Potonoro

Gambar 4. 2 Gambar Kamar 2

3. Kamar 3

   Panti Jompo Madania Potonoro

Gambar 4. 3 Gambar Kamar 3

4. Kamar 4

   Panti Jompo Madania Potonoro

Gambar 4. 4 Gambar Kamar 4

5. Kamar 5

   Biodata Lansia kamar 5:

   Nama : RIMATINA

   TTL : Semarang, 26 Agustus 1946

   Umur : 76 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 5:


Gambar 4. 5 Gambar Kamar 5

6. Kamar 6

   Nama : Putu Sukarma

   TTL : Tabanan, 21 Januari 1944

   Umur : 78 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 6:

Gambar 4. 6 Gambar Kamar 6

7. Kamar 7

   Nama : Suparti

   TTL : Sragen, 5 Juli 1960

   Umur : 62 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 7:


Gambar 4. 7 Gambar Kamar 7

8. Kamar 8

   Nama : Stien Margotje

   TTL : 26 Maret 1949

   Umur : 73 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 8:


Gambar 4. 8 Gambar Kamar 8

9. Kamar 9

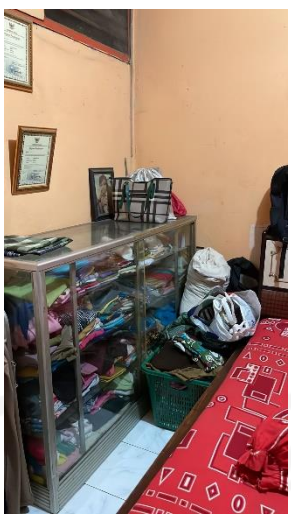
   Nama : Yeda Lamiung

   TTL : 01 Januari 1953

   Umur : 68 Tahun

   Jenis Kelamin : Perempuan

   Berikut merupakan gambar kamar 9:




Gambar 4. 9 Gambar Kamar 9

10. Kamar 10

Nama : Sularmi

TTL : Karanganyar, 1 Januari 1950

Umur : 72 Tahun

Jenis Kelamin : Perempuan

Berikut merupakan gambar kamar 10:


Gambar 4. 10 Gambar Kamar 10

4.2.2 Data Produk Pada Kamar Lansia

Berdasarkan data kamar yang telah didapatkan, berikut merupakan jenis-jenis produk atau peralatan yang ada pada kamar tidur atau tempat istirahat lansia.

Tabel 4. 1 Data Jenis Produk di Kamar Lansia

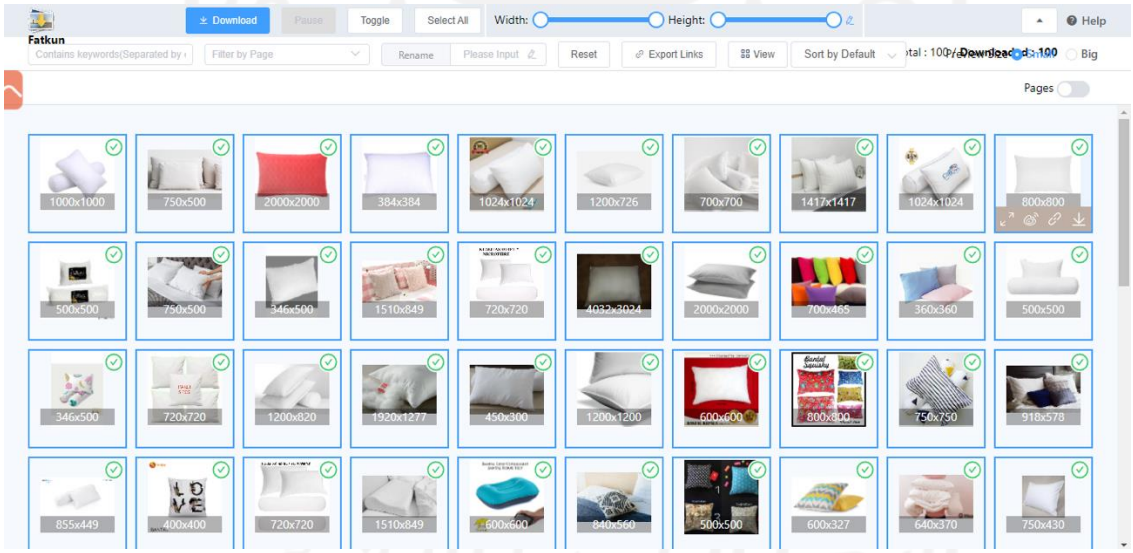
Kamar	Jenis Produk
1	Bantal, Guling, Kain, Speaker Home Theater, Lampu Meja, Sisir, Jam Dinding, Lemari, Kursi, Koper.
2	Radio, Bantal, Selimut, Kacamata, Bingkai Foto, Kain, Gunting.
3	Jam Dinding, Bantal, Kursi, Botol, Lemari, Spons Mandi, Galon, Sapu Lidi.
4	Bantal, Guling, Radio, Botol, Jam Dinding, Termos, Keset Kaki, Kipas Angin, Gelas.
5	Bantal, Kain, Kipas Angin, Meja, Kursi, Korek, Cotton Buts, Botol, Kalendar.

Kamar	Jenis Produk
6	Bantal, Guling, Kain, Lemari, Koper.
7	Bantal, Guling, Meja, Kipas, Sisir, Kipas.
8	Bantal, Kipas, Meja Lemari.
9	Kursi, Meja, Etalase, Lemari, Tas, Etalase, Guling, Bantal.
10	Bantal, Guling, Jam Dinding, Lemari, TV, Tas

Berdasarkan data keseluruhan produk dari setiap kamar, produk atau peralatan yang akan digunakan pada penelitian kali ini adalah: Bantal, Guling, Lampu Meja, Sisir, Jam Dinding, Lemari, Kursi, Koper, Radio, Selimut, Kacamata, Bingkai Foto, Gunting, Botol, Spons Mandi, Galon, Sapu Lidi, Termos, Keset Kaki, Kipas Angin, Gelas, Korek, *Cotton Buds*, Kalendar, dan Tas.

4.2.3 Pengumpulan data *image* setiap produk/peralatan.

Berdasarkan data produk yang telah didapatkan, selanjutnya adalah pengumpulan data *image* yang diambil melalui *crawling* dari *google image* dengan bantuan *extensions* pada *google chrome* yaitu *Fatkun Batch Download Image*.



Gambar 4. 11 Pengumpulan *Dataset*

4.3 *Pre-Processing Data*

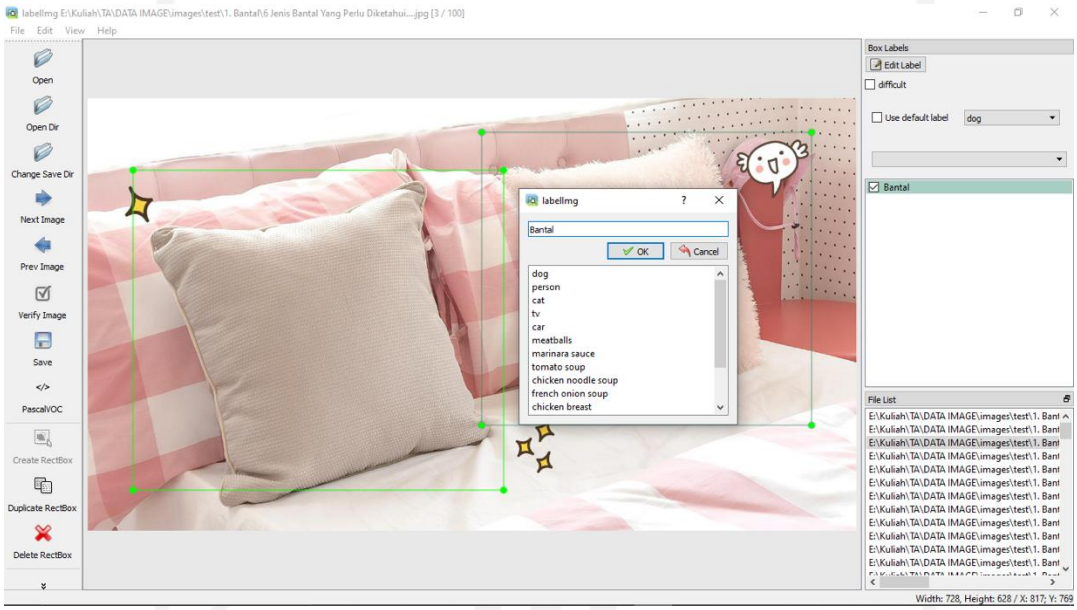
4.3.1 Pelabelan *Dataset*

Pelabelan data merupakan *step* awal dalam tahapan *pre-processing* data, dimana *dataset* akan diberi label sesuai dengan *class* objek pada setiap gambar. Data gambar dari setiap produk yang digunakan pada penelitian ini akan diberikan label satu persatu sesuai dengan *class* objek tersebut dengan bantuan *library LabelImg*. untuk menjalankan *library* tersebut perintah yang digunakan pada *Anaconda Prompt* (*Anaconda3*) sebagai berikut:

```
python labelimg.py
```

Gambar 4. 12 Perintah Menjalankan *Library labelimg*

Pelabelan gambar ini bertujuan untuk menyimpan seluruh informasi gambar yang selanjutnya disimpan dalam *file* ".XML" dengan format *PASCAL VOC*, format yang digunakan oleh ImageNet. Setiap produk atau peralatan yang digunakan pada penelitian ini memiliki jumlah gambar sebesar 100 data gambar yang akan dibagi menjadi data *training* dan data *testing* dengan perbandingan 80% untuk data *training* dan 20% untuk data *testing*. Berikut merupakan visualisasi dari pemberian label pada *library LabelImg*.



Gambar 4. 13 Proses Pelabelan *Dataset*

4.3.2 Konversi Dataset XML ke csv

Setelah melakukan proses pelabelan data menggunakan *LabelImg*, didapatkan hasil *output* berupa *file* dengan format .XML (*Xtensible Markup Language*) yang akan dikonversi menjadi .csv, dimana hal ini dilakukan untuk konversi dataset menjadi *file TFRecord*. Untuk mengkonversi *file* dengan format *XML* menjadi *file csv* terdapat beberapa *pseudocode* dalam melakukan konversi *xml to scv* seperti pada tabel 4.2

Tabel 4. 2 *Pseudocode* Proses Konversi XML to csv

Pseudocode XML to csv	
Pseudocode	Fungsi

def xml_to_csv(path):
xml_list = []
for xml_*file* in glob.glob(path + '/*.xml'):
tree = ET.parse(xml_*file*)
root = tree.getroot()
for member in root.findall('object'):
value = (root.find('*file*name').text,
int(root.find('size')[0].text),
int(root.find('size')[1].text),
member[0].text,
int(member[4][0].text),
int(member[4][1].text),
int(member[4][2].text),
int(member[4][3].text)
)
	Mendefinisikan fungsi dari *xm_to_csv* dan mengatur objek dengan memberikan label "*tree*" yang akan diteruskan pada "*xml_file*". Lalu mendefenisikan value yang akan dibutuhkan dalam *file xml* terkait informasi pada *bounding box* dalam gambar.

xml_list.append(value) column_name = ['*file*name', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax'] xml_df = pd.DataFrame(xml_list, columns=column_name) return xml_df	Menambahkan nilai *value* pada *xml_list* sesuai dengan urutan pada *coloum_name* dan mendefinisikan *xml_df* dengan struktur data dengan urutan *xml_list* hingga *column_name*

untuk menjalankan *script* tersebut perintah yang digunakan pada *Anaconda Prompt* (*Anaconda3*) sebagai berikut:

```
python xml_to_csv.py
```

Gambar 4. 14 Perintah Menjalankan Konversi *XML to csv*

Setelah menjalankan *script* tersebut dan berhasil dilakukannya konversi *xml to csv*, maka didapatkan 2 *file csv* yaitu *file test_labels.csv* dan *train_labels.csv* yang mana *file* tersebut akan dikonversi lagi menjadi *file TFRecord*, berikut merupakan hasil konversi *file csv* yang didapatkan:

	A	B	C	D	E	F	G	H
1	filename	width	height	class	xmin	ymin	xmax	ymax
2	Jual Bingk	1000	1000	Bingkai Foto	417	491	730	658
3	Jual Bingk	1000	1000	Bingkai Foto	257	554	394	652
4	Jual Bingk	1000	1000	Bingkai Foto	216	358	407	541
5	Jual Bingk	1000	1000	Bingkai Foto	110	317	210	458
6	Jual Bingk	1000	1000	Bingkai Foto	234	204	366	354
7	Jual Bingk	1000	1000	Bingkai Foto	397	247	549	361
8	Jual Bingk	1000	1000	Bingkai Foto	567	202	779	375
9	Jual Bingk	1000	1000	Bingkai Foto	784	317	886	452
10	Jual Bingk	381	360	Bingkai Foto	165	179	318	303
11	Jual Bingk	381	360	Bingkai Foto	17	198	139	295
12	Jual Bingk	381	360	Bingkai Foto	4	10	138	170
13	Jual Bingk	500	500	Bingkai Foto	19	120	227	375
14	Jual Bingk	500	500	Bingkai Foto	312	288	479	484
15	Jual Bingk	500	500	Bingkai Foto	306	59	476	264
16	Jual Bingk	800	800	Bingkai Foto	243	64	744	740
17	Jual Bingk	800	800	Bingkai Foto	353	139	783	691
18	Jual Bingk	700	700	Bingkai Foto	203	218	481	502
19	Jual Custo	1024	1024	Bingkai Foto	165	127	891	948
20	Jual Jam D	600	600	Jam Dinding	19	14	596	586
21	Jual Jam D	599	597	Jam Dinding	58	83	549	543
22	Jual Jam D	360	360	Jam Dinding	88	33	255	195
23	Jual Jam D	360	360	Jam Dinding	29	30	350	334
24	Jual JAM D	500	500	Jam Dinding	44	75	424	425

Gambar 4. 15 Hasil Konversi *File csv*

Pada hasil konversi *XML* menjadi *csv* menghasilkan *output* seperti gambar 4.15. *Output* tersebut berisi kolom "*filename, width, height, class, xmin, ymin, xmax,* dan *ymax*", dimana *filename* merupakan nama *file* pada *dataset* (gambar), lalu *width* dan *height* merupakan ukuran (lebar dan tinggi) dari setiap *dataset* (gambar) yang digunakan. Pada kolom *class* merupakan hasil dari pelabelan *dataset* pada setiap objek dari setiap gambar yang dilakukan pada pelabelan *dataset* menggunakan *labelimg*. Selanjutnya kolom *xmin* dan *ymin* merupakan nilai minimal *pixel* pada *width* dan *height*, sedangkan *xmax* dan *ymax* merupakan nilai maximal *pixel* pada *width* dan *height* dari kotak atau *box* yang digunakan saat menandai sebuah objek pada gambar yang dilakukan pada proses pelabelan gambar menggunakan *labelimg* agar sesuai dengan objek.

4.3.3 Konversi *Dataset cvs* menjadi *TFRecord*

Setelah melakukan proses konversi *XML* menjadi *csv*, tahapan selanjutnya melakukan konversi *dataset csv* menjadi *TFRecord* atau *Tensorflow Record*, dimana *file TFRecord* merupakan akan menjadi data input yang akan dibaca oleh *tensorflow* atau yang biasa disebut dengan *feeding data*. Dalam proses *feeding data*, informasi dataset yang telah tersimpan dalm format *TFrecord*. Untuk mengkonversi *file* dengan format *csv* menjadi *file TFRecord*, terdapat beberapa pseudocode untuk menjalankan proses konversi *csv* menjadi *TFRecord* seperti pada tabel 4.3

Tabel 4. 3 *Pdeudocode* Proses Konversi *csv to TFRecord*

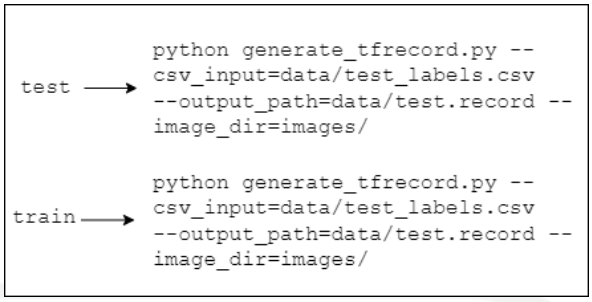
Pseudocode Generate TFRecord	
Pseudocode	Fungsi
flags = tf.compat.v1.flags flags.DEFINE_string('csv_input', '', 'Path to the CSV input') flags.DEFINE_string('output_path', '', 'Path to output TFRecord') flags.DEFINE_string('image_dir', '', 'Path to images') FLAGS = flags.FLAGS	Mendefinisikan *flag* yang nilainya dapat berupa string atau kata

def class_text_to_int(row_label):
  if row_label == 'Bantal':
  return 1
  if row_label == 'Bingkai Foto':
  return 2
	Mendefinisikan *label* atau kelas sesuai dengan *label map* yang telah dibuat

Pseudocode Generate TFRecord

Pseudocode	Fungsi
tf_example = tf.train.Example(features=tf.train.Features(feature={ 'image/height': dataset_util.int64_feature(height), 'image/width': dataset_util.int64_feature(width), 'image/*file*name': dataset_util.bytes_feature(*file*name), 'image/source_id': dataset_util.bytes_feature(*file*name), 'image/encoded': dataset_util.bytes_feature(encoded_jpg), 'image/format': dataset_util.bytes_feature(image_format), 'image/object/bbox/xmin': dataset_util.float_list_feature(xmins), 'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs), 'image/object/bbox/ymin': dataset_util.float_list_feature(ymins), 'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs), 'image/object/class/text': dataset_util.bytes_list_feature(classes_text), 'image/object/class/label': dataset_util.int64_list_feature(classes), })) return tf_example	Mendefinisikan "*example*" dimana *example* adalah *proto* standar yang menyimpan data untuk pelatihan dan inferensi.
writer.close() output_path = os.path.join(os.getcwd(), FLAGS.output_path) print('Successfully created the TFRecords: {}'.format(output_path))	Mendefinisikan output dan memerintahkan untuk menampilkan teks "Successfully created the TFRecords" apabila proses Generate TFRecord telah selesai

Untuk menjalankan *script* tersebut, perintah yang digunakan pada *Anaconda Prompt* (*Anaconda3*) sebagai berikut:



Gambar 4. 16 Perintah Menjalankan Konversi *csv to TFRecord*

4.3.4 Konfigurasi *Label Map*

Langkah terakhir pada tahap *pre-processing* adalah konfigurasi *label map*, dimana konfigurasi *label map* merupakan proses pemetaan label atau *class* yang digunakan dalam pemberian nama objek yang akan dideteksi. Pada penelitian ini akan dilakukan pemetaan label sebanyak 25 objek yang akan dideteksi, sehingga pada *label map* ini juga akan terdapat 25 item *label*. Pada proses konfigurasi label map ini terdiri dari nomor id dan nama dari setiap benda atau objek. Label map ini akan disimpan dalam format "*pbtxt*" yang mana akan digunakan untuk proses perancangan *model*. Berikut merupakan konfigurasi dari label map.

```
item {
        id 1
        name: 'Bantal'
}
item {
        id 2
        name: 'Bingkai Foto'
}
```

Gambar 4. 17 Konfigurasi *Label Map*

4.4 Perancangan *Model*

Setelah melakukan tahapan *pre-processing*, selanjutnya akan melakukan perancangan *model* atau sistem dari *object detection*. Pada penelitian ini, konfigurasi yang digunakan adalah konfigurasi *ssd mobilenet v2 fpnlite 640x640 coco17_tpu-8*. Pada penelitian ini, konfigurasi *pipeline* dilakukan dengan beberapa parameter.

Pada konfigurasi *pipeline*, dilakukan pendefinisian jumlah kelas pada *script num_classes:* 25 yang artinya jumlah kelas berjumlah 25 sesuai dengan jumlah kelas yang telah dibuat pada *label map*, lalu konfigurasi selanjutnya adalah pendefinisian *batch_size*: 5 yang artinya jumlah sampel dari data yang akan disebarkan ke dalam *neural network* berjumlah 2 yang mana sampel tersebut akan diambil secara random dari semua *sample train dataset*, selanjutnya pendefinisian *num_steps*: 50000 yang artinya jumlah langkah

maksimal dalam proses *train* adalah sebesar 50000 *steps*, dan pendefinisian terakhir pada *pipeline* yaitu *num_eval_steps*: 25000 yang artinya jumlah langkah maksimal dalam proses *eval* adalah sebesar 25000 *steps*. Adapun *pseudocode* yang digunakan dalam melakukan konfigurasi atau perancangan *model* sebagai berikut:

Tabel 4. 4 *Pdeudocode* Proses Konfigurasi *Model*

Pseudocode Konfigurasi *Model*	
Pseudocode	Fungsi

model {
ssd {
inplace_batchnorm_update: true
freeze_batchnorm: false
num_classes: 25
box_coder {
faster_rcnn_box_coder {
y_scale: 10.0
x_scale: 10.0
height_scale: 5.0
width_scale: 5.0
}
}
	Menentukan jumlah kelas sesuai dengan jumlah kelas atau *label* dan kode box *faster rcnn* dengan skala x, y, *height* dan *width*.

train_config: {
fine_tune_checkpoint_version: V2
fine_tune_checkpoint:
"/content/gdrive/MyDrive/ObjectDetection/saved_*model*/ckpt-0"
fine_tune_checkpoint_type: "detection"
batch_size: 5
sync_replicas: true
startup_delay_steps: 0
replicas_to_aggregate: 8
num_steps: 50000
data_augmentation_options {
random_horizontal_flip {
}
}
	Menentukan *path* atau jalur dari "*fine_tune_checkpoint*", lalu mengubah *type* dari "*classification*" menjadi "*detection*" dan menentukan jumlah *batch_size* dan jumlah *num_steps*

train_input_reader: {
label_map_path:
"/content/gdrive/MyDrive/ObjectDetection/data/label_map.pbtxt"
tf_record_input_reader {
input_path:
"/content/gdrive/MyDrive/ObjectDetection/data/train.record"
}
}
	Menentukan *path* atau jalur dari *input* proses *training* dari *label map* dan *Train TFRecord*

Pseudocode Konfigurasi *Model*	
Pseudocode	Fungsi

eval_input_reader: {
label_map_path:
"/content/gdrive/MyDrive/ObjectDetection/data/label_map.pbtxt"
shuffle: false
num_epochs: 1
tf_record_input_reader {
input_path:
"/content/gdrive/MyDrive/ObjectDetection/data/test.record"
}
}
	Menentukan *path* atau jalur dari input proses *eval/testing* dari *label map* dan *Test TFRecord*

4.5 *Training Model*

Setelah melakukan *konfigurasi pipeline* atau perancangan *model*, selanjutnya akan masuk pada tahapan dari *neural network*, tahapan utama dari proses *neural network* adalah *training model*, dimana semua *dataset* akan dilatih atau di-*training* untuk mengenali dan mempelajari 25 pola *object*. Semua proses yang terdapat pada jaringan CNN akan bekerja pada proses *training model*. Pada *training model* akan didapatkan output berupa *model* yang dapat mendeteksi objek dengan tingkat akurasi yang tinggi.

Pada proses *training model,* dilakukan dengan bantuan *Google Colaboratory* untuk melakukan *training model* dengan memanfaatkan fasilitas GPU yang ada. Proses *training* dimulai dengan menjalankan perintah *training* sebagai berikut:

```
!python /content/models/research/object_detection/model_main_tf2.py \
    --pipeline_config_path={pipeline_config_path} \
    --model_dir={model_dir} \
    --alsologtostderr \
    --num_train_steps={num_steps} \
    --sample_1_of_n_eval_examples=1 \
    --num_eval_steps={num_eval_steps}
```

Gambar 4. 18 Perintah *Training Model*

Gambar 4.18 merupakan perintah untuk menjalankan proses *training model* pada *script model_main_tf2.py*. Untuk melakukan proses *training model*, terdapat beberapa *pseudocode* yang digunakan sebagai berikut:

Tabel 4. 5 *Pdeudocode* Proses *Train Model*

Pseudocode Train Model	
Pseudocode	Fungsi
flags.DEFINE_string('pipeline_config_path', None, 'Path to pipeline config ' '*file*.') flags.DEFINE_integer('num_train_steps', None, 'Number of train steps.') flags.DEFINE_bool('eval_on_train_data', False, 'Enable evaluating on train ' 'data (only supported in distributed training).') flags.DEFINE_integer('sample_1_of_n_eval_examples', None, 'Will sample one of ' 'every n eval input examples, where n is provided.') flags.DEFINE_integer('sample_1_of_n_eval_on_train_examples', 5, 'Will sample ' 'one of every n train input examples for evaluation, ' 'where n is provided. This is only used if ' '`eval_training_data` is True.')	Mendefinisikan flag yang nilainya dapat berupa "string/kata", "integer/bilangan bulat", dan "boolean/tipe data(true & false)"
def main(unused_argv): flags.mark_flag_as_required('*model*_dir') flags.mark_flag_as_required('pipeline_config_path') tf.config.set_soft_device_placement(True)	Memastikan bahwa flag bukan None/Null selama eksekusi program

Pemantauan proses *training model* dapat dilakukan menggunakan *module* yang tersedia pada *tensorflow* yaitu *Tensorboard*. Penggunaan *tensorboard* dilakukan dengan bantuan *google colaboratory*, berikut merupakan perintah yang digunakan untuk menggunakan *tensorboard*:

```
%reload_ext tensorboard
%tensorboard --logdir '/content/saved_model/train'
```

Gambar 4. 19 Perintah *Tensorboard*

Setelah menjalankan perintah untuk menggunakan *tensorboard* pada gambar 4.19 diatas, *tensorboard* akan secara otomatis memanggil *file checkpoint* berupa grafik yang dihasilkan selama proses *training*. Pada *tensorboard* peneliti dapat memantau grafik-grafik yang ada ketika proses *training* sedang berlangsung. Grafik tersebut antara lain: Grafik *Loss* (*classification_loss, localization_loss, regularization loss,* dan *total_loss*), Grafik *Learning Rate,* dan Grafik *Steps per second*. Berikut merupakan *interface* dari *tensorboard*:

Gambar 4. 20 *Interface Tensorboard*

4.6 *Export Model Inference Graph*

Proses *training* yang dilakukan akan menghasilkan *file checkpoint* yang dibuat secara otomatis oleh *Tensorflow* berbentuk *graph tensor* yang bertujuan untuk menyimpan atau me-*record* hasil dari proses *training* yang telah dilakukan. Setelah melakukan proses *training*, langkah selanjutnya adalah melakukan *export model* yang telah di-*training* untuk mendapatkan *model* yang siap untuk digunakan. Berikut merupakan perintah yang digunakan untuk melakukan *export model*:

```
!python /content/models/research/object_detection/exporter_main_v2.py \
    --trained_checkpoint_dir {model_dir} \
    --output_directory {output_directory} \
    --pipeline_config_path {pipeline_config_path}
```

Gambar 4. 21 Perintah *Export Model*

Untuk menjalankan proses dari *export model*, terdapat beberapa pseudocode yang digunakan pada proses ini, diantaranya seperti tabel

Tabel 4. 6 *Pdeudocode* Proses *Export Model*

Pseudocode Export Model	
Pseudocode	Fungsi
flags.DEFINE_string('input_type', 'image_tensor', 'Type of input node. Can be ' 'one of [`image_tensor`, `encoded_image_string_tensor`, ' '`tf_example`, `float_image_tensor`, ' '`image_and_boxes_tensor`]') flags.DEFINE_string('pipeline_config_path', None, 'Path to a pipeline_pb2.TrainEvalPipelineConfig config ' '*file*.') flags.DEFINE_string('trained_checkpoint_dir', None, 'Path to trained checkpoint directory') flags.DEFINE_string('output_directory', None, 'Path to write outputs.') flags.DEFINE_string('config_override', '', 'pipeline_pb2.TrainEvalPipelineConfig ' 'text proto to override pipeline_config_path.') flags.DEFINE_boolean('use_side_inputs', False, 'If True, uses side inputs as well as image inputs.')	Mendefinisikan flag yang nilainya dapat berupa "string/kata", "integer/bilangan bulat", dan "boolean/tipe data(true & false)"
flags.mark_flag_as_required('pipeline_config_path') flags.mark_flag_as_required('trained_checkpoint_dir') flags.mark_flag_as_required('output_directory')	Memastikan bahwa flag bukan None/Null selama eksekusi program

Setelah proses *export model* telah selesai dijalankan, maka hasil yang didapatkan adalah *file model* dalam bentuk format *protobuf* ".pb". *file model* tersebut merupakan final *file* yang akan digunakan untuk uji coba pada *images* agar dapat mengetahui tingkat akurasi yang didapatkan oleh *model* tersebut.

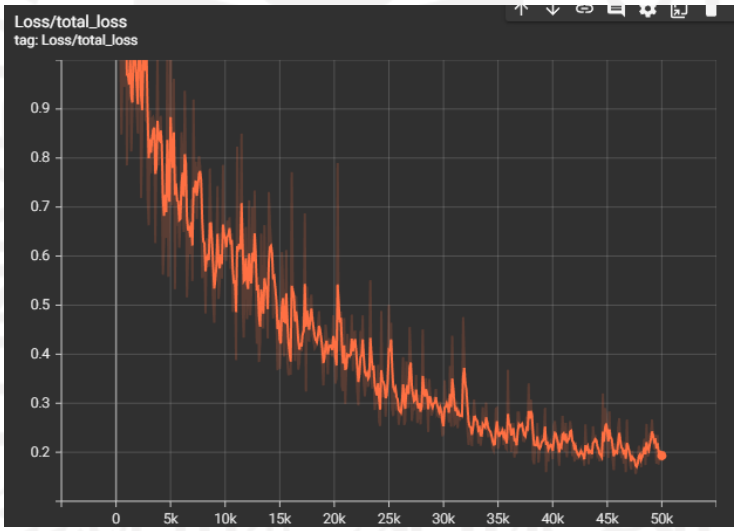
BAB V
PENGUJIAN SISTEM DAN PEMBAHASAN

5.1 *Model* Hasil *Training*

Proses *training* dilakukan dengan jumlah *steps* sebanyak 50000 dan jumlah *batch* sebanyak 5, dimana proses *training* ini membutuhkan waktu selama 5 jam 48 menit. Pada proses *training*, seluruh proses *train* akan terekam pada *tensorboard* seperti *training loss, learning rate,* dan *steps per sec* yang akan divisualisasikan dalam bentuk sebuah grafik. Berikut merupakan hasil yang didapatkan selama melakukan *training model*.

5.1.1 *Total Loss*

Hasil dari nilai *total loss* merupakan nilai *error* yang dihasilkan selama dari proses *training*. Hasil *error* yang dihasilkan dari setiap *training* akan direkam dan divisualisasikan kedalam grafik. Berikut merupakan hasil dari *total loss* yang didapatkan selama proses *training*:

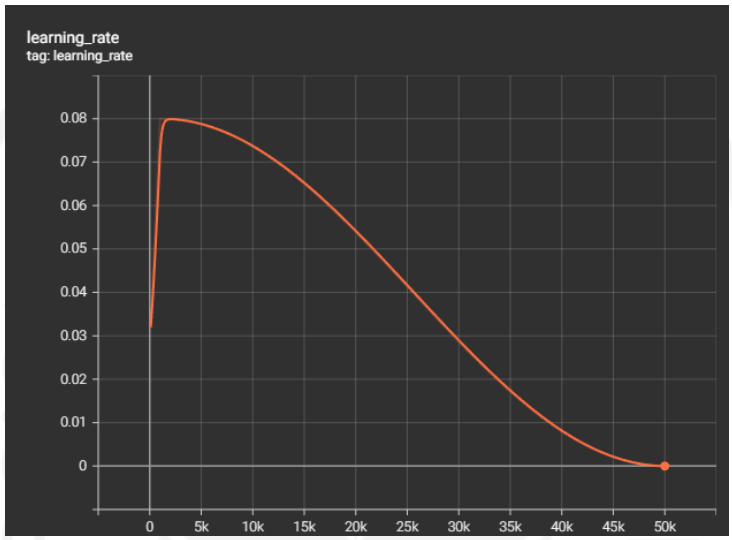


Gambar 5. 1 Grafik *Total Loss*

Berdasarkan grafik diatas menunjukkan bahwa adanya penurunan nilai *total loss,* dimana pada saat awal *steps* memiliki nilai *total loss* yang sangat tinggi yaitu sebesar 1.885. Namun semakin banyaknya *steps* yang dilakukan, nilai dari *total loss* semakin menurun. Hal ini terlihat dari nilai *total loss* pada *steps* 50000 sebesar 0.1813. Dengan hal ini, menunjukkan bahwa semakin kecil nilai *total loss* yang didapatkan pada saat proses *training*, maka *model* yang digunakan untuk mendeteksi objek akan semakin baik.

5.1.2 *Learning Rate*

Learning rate merupakan salah satu parameter *training* untuk menghitung nilai koreksi bobot pada waktu proses *training*. Nilai *learning rate* menunjukkan nilai kecepatan belajar dari jaringan, Nilai *learning rate* berada pada *range* nol (0) hingga satu (1), dimana semakin besar nilai *learning rate* maka proses *training* yang dilakukan akan berjalan semakin cepat (Syarifah, 2018).



Gambar 5. 2 Grafik *Learning Rate*

Berdasarkan hasil grafik diatas, semakin banyaknya jumlah *steps* yang dilakukan pada proses *training*, maka nilai *learning rate* yang digunakan semakin kecil, sehingga proses training membutuhkan waktu semakin lama. Hal ini disebabkan karena apabila nilai *learning rate* yang digunakan semakin kecil maka *epoch* yang dibutuhkan akan semakin banyak untuk mencapai nilai target yang diinginkan.

5.2 Hasil Pendeteksian Objek pada *Images*

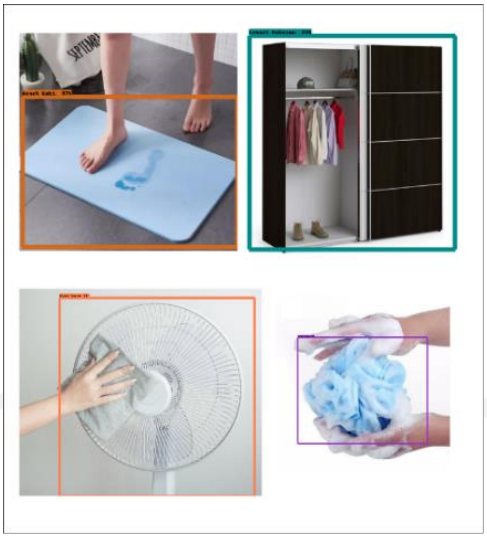
Setelah melakukan *export model* untuk mendapatkan *model* akhir, selanjutnya *model* tersebut akan dilakukan pengujian untuk mengetahui tingkat akurasi dari *model* tersebut.

Untuk melakukan pengujian *model* yang telah di-*export* menjadi dalam format *protobuf*, selanjutnya *model* tersebut akan dilakukan pengujian terhadap gambar, dimana untuk melakukan pengujian tersebut, digunakan beberapa *pseudocode* seperti tabel 5.1.

Tabel 5. 1 *Pseudocode* Pengujian *Model*

Pseudocode Pengujian *Model*	
Pseudocode	Fungsi
tf.keras.backend.clear_session() *model* = tf.saved_*model*.load(f'//content/gdrive/MyDrive/ObjectDetection/inference_graph/saved_*model*')	Mereset semua status yang dihasilkan oleh Keras dari *model* yang akan di load atau digunakan pada proses pengujian *model*
category_index = label_map_util.create_category_index_from_labelmap('/content/gdrive/MyDrive/ObjectDetection/data/label_map.pbtxt', use_display_name=True)	Memuat atau load label map yang telah dibuat.
for image_path in glob.glob('/content/gdrive/MyDrive/ObjectDetection/data/images/Test_Images/*.jpg'): image_np = load_image_into_numpy_array(image_path) output_dict = run_inference_for_single_image(*model*, image_np)	Melakukan load image yang akan diuji coba pada sebuah folder dengan format gambar berupa .jpg
vis_util.visualize_boxes_and_labels_on_image_array(image_np, output_dict['detection_boxes'], output_dict['detection_classes'], output_dict['detection_scores'], category_index, instance_masks=output_dict.get('detection_masks_reframed', None), use_normalized_coordinates=True, line_thickness=8)	Memvisualisasikan image yang akan dideteksi, mulai dari detection box untuk mengidentifikasi objek, lalu kelas dari objek yang telah teridentifikasi dan tingkat akurasi yang didapatkan
display(Image.fromarray(image_np))	Menampilkan hasil pengujian *model* terhadap gambar

Model akhir akan dilakukan pengujian pada *images* dengan bantuan *google colaboratory*, berikut merupakan hasil dari *testing* atau pengujian dari *model* terhadap *images*.



Gambar 5. 3 Hasil Pengujian *Model* Pada *Images*

Pengujian *model* terhadap *images* dilakukan untuk semua *class* atau *label* yang telah dibuat, pada setiap *class* dilakukan pengujian dengan masing-masing 5 *images* dan mencari nilai rata-rata dari setiap pengujian. Berikut merupakan hasil tingkat akurasi yang didapatkan dari pengujian *model* pada setiap *class*:

Tabel 5. 2 Hasil Tingkat Akurasi

Nama Produk	Rata-Rata Akurasi
Bantal	90%
Guling	91%
Sisir	92%
Jam Dinding	98%
Lemari Pakaian	95%
Kursi	96%
Koper	93%
Radio	96%
Selimut	94%
Kacamata	94%
Bingkai	94%
Gunting	96%
Botol	95%
Lampu Meja	95%
Spons Mandi	95%
Galon	93%
Sapu Lidi	95%
Termos	98%
Keset Kaki	96%
Kipas Angin	97%
Korek Gas	95%
Cotton Buds	96%

Nama Produk	Rata-Rata Akurasi
Kalendar	95%
Gelas	96%
Tas	96%
Rata-Rata	95%

Berdasarkan hasil nilai tingkat akurasi yang diujikan pada 5 *images* dari setiap *class*, didapatkan nilai rata-rata dari hasil nilai tingkat akurasi pada *model* yang dibuat sebesar 95%.

5.3 Rekomendasi Sistem Pada Lansia

Proses menua dapat menimbulkan berbagai macam masalah, baik masalah secara fisik, biologis, mental maupun masalah sosial ekonomi, contohnya adalah perubahan fisik yang terlihat sangat jelas seperti gangguan penglihatan pada lanjut usia (lansia). Penurunan penglihatan adalah keluahan besar yang dialami oleh lanjut usia dikarenakan persepsi terhadap lingkungan yang memiliki hubungan dengan rasa aman yang dialami oleh lanjut usia.

Dengan banyaknya aktivitas yang dilakukan oleh lansia, terkhususnya di ruang istirahat atau tempat tidur dan juga penurunan penglihatan yang dapat menjadi penyebab dari peningkatan resiko yang berbahaya bagi lanjut usia seperti jatuh, salah mengambil barang atau alat yang berbahaya dll.

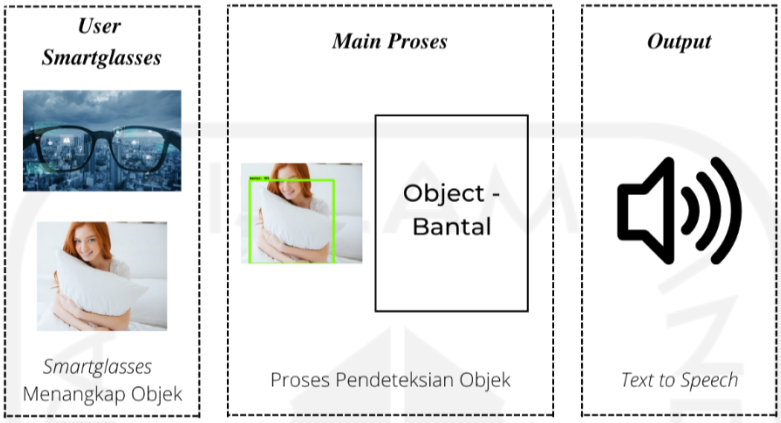
Berdasarkan penelitian yang dilakukan oleh Nugroho (2008) dalam Nizar Argyatiyasa, Suprajitno, dan Wiwin Martiningsih (2015) beranggapan lansia masih tidak menyadari pentingnya upaya pencegahan dari terjadinya kecelakaan, dimana lansia seharusnya menggunakan alat pengaman seperti kacamata dan tongkat yang dapat membantu dalam beraktivitas sehari-hari.

Oleh karena itu untuk mencegah dari potensi resiko tersebut, dibutuhkan suatu alat bantu yang dapat membantu pemahaman penglihatan bagi lanjut usia. Salah satu langkah awal yang dapat menjadi cara agar dapat meminimalisir resiko-resiko tersebut adalah dengan sistem pendeteksi objek.

Dimana dengan adanya sistem pendeteksi objek, dapat menjadi langkah awal untuk menciptakan berbagai produk atau alat bantu yang dapat membantu lansia dalam pengelihatan atau pemahaman disekitarnya.

Salah satu produk yang dapat digunakan sebagai implementasi dari sistem *object detection* adalah dengan menciptakan alat bantu berupa kacamata pintar (*smartglasses*) yang dapat membantu lansia dalam mendeteksi objek di daerah sekitarnya dan juga dapat

menjadi upaya dalam pencegahan terjadinya kecelakaan. Pada penelitian ini, telah dibuat sebuah sistem *object detection*, dimana sistem tersebut dapat digunakan sebagai *main* proses dalam pengenalan *object* pada *smart glasses*. Berikut merupakan ilustrasi dari penerapan sistem *object detection* pada *smartglasses*:



Gambar 5. 4 Ilustrasi *Object Detection* Pada *Smartglasses*

Berdasarkan penelitian yang telah dilakukan oleh Mukhriddin Mukhiddinov dan Jinsoo Cho pada tahun 2021 terkait "*Smart Glass System Using Deep Learning for the Blind and Visually Impaired*", didapatkan hasil bahwa sistem *object detection* dapat diterapkan pada sebuah *smart glass* yang dapat membantu *user* atau pengguna untuk mengetahui *object* yang ada pada daerah sekitar dan dapat ditransformasikan menjadi sebuah audio dengan menggunakan *text to speech*. Dengan adanya penelitian tersebut, Hasil dari penelitian tersebut dapat digunakan sebagai acuan dalam penelitian selanjutnya untuk melanjutkan penelitian ini dengan mengembangkan sistem dan mengimplementasikan pada *smartglasses* yang dapat membantu lanjut usia untuk membantu pemahaman peralatan di daerah tempat tidur.

5.4 Evaluasi *Model*

Berdasarkan hasil dan pengolahan data yang telah dilakukan, didapatkan hasil berupa *model* yang terbentuk dari proses *training* selama 5 jam 48 menit, dimana pada proses ini dibutuhkan waktu yang cukup lama karena menggunakan spesifikasi perangkat seperti *Randim Access Memory* (RAM) dan *Graphic Processing Unit* (GPU) yang rendah walaupun menggunakan jumlah *steps* dan jumlah *batch* yang terbilang kecil yaitu sebesar 50000 *steps* dan *batch size* 5. Untuk meningkatkan performa atau tingkat akurasi yang dihasilkan dapat dilakukan dengan menambah jumlah *steps* dan *batch size* hingga 64 atau 128, dan untuk meminimalisir waktu proses *training*, peneliti harus menggunakan spesifikasi perangkat yang lebih tinggi.

Dataset yang digunakan oleh peneliti dalam penelitian ini sangat terbatas dikarenakan menggunakan teknik *crawling* pada *google images* dengan bantuan *tools* Fatkun Batch Downloader, sehingga *dataset* yang didapatkan random dan beberapa *dataset* yang setiap objeknya hanya menampilkan 1 sampai 2 sisi saja, sehingga pembacaan yang didapatkan sehingga berpengaruh pada *model* yang didapatkan. Untuk mengatasi hal tersebut, penelitian selanjutnya dapat melakukan pengumpulan *dataset* dari segala sisi setiap objek yang akan digunakan. Dan juga ragam dari *dataset* yang digunakan masih minim, dimana pada penelitian ini hanya menggunakan 25 produk yang ada pada daerah kamar tidur dari lansia, sehingga apabila *model* ini digunakan untuk mendeteksi objek pada kamar tidur lansia di tempat umum, memungkinkan *model* tidak dapat mendeteksi objek apabila objek tersebut tidak terdapat dalam *dataset objek*, oleh karena itu penelitian selanjutnya dapat menambahkan jumlah *dataset* dan juga ragam atau jenis dari objek pada gambar untuk melatih *model* yang akan dibangun agar dapat mencapai akurasi yang lebih tinggi dan maksimal.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan *model* dan analisis yang telah dilakukan, maka didapatkan beberapa kesimpulan sebagai berikut:

1. *Model* yang digunakan dalam penelitian ini adalah *pre-trainined model ssd mobilenet v2 fpnlite 640x640*. Sehingga *model* yang terbentuk berdasarkan hasil proses *training* memiliki jumlah *steps* sebanyak 50.000 dengan 5 *batch size* serta memiliki nilai *total loss* yang rendah sebesar 0.1813. Dengan hal ini, menunjukkan bahwa *model* yang digunakan untuk mendeteksi objek sangat baik.
2. Hasil dari pendeteksian produk pada daerah tempat istirahat atau tempat tidur lanjut usia (lansia) dengan menggunakan metode *convolutional neural network* (CNN) memiliki rata-rata tingkat akurasi yang sangat tinggi yaitu sebesar 95%.
3. Untuk mencegah terjadinya potensi resiko berbahaya, dibutuhkan suatu alat bantu yang dapat membantu pemahaman penglihatan bagi lanjut usia. Salah satu langkah awal yang dapat menjadi cara agar dapat meminimalisir resiko-resiko tersebut adalah dengan sistem pendeteksi objek, dimana dengan adanya sistem pendeteksi objek, dapat menjadi langkah awal untuk menciptakan berbagai inovasi produk atau alat bantu yang dapat membantu lansia dalam pengelihatan atau pemahaman disekitarnya. Salah satunya adalah *smartglasses* atau kacamata pintar sebagai alat bantu dan juga sebagai upaya dalam pencegahan terjadinya kecelakaan. Sehingga konsep penerapan *object detection* dapat dilihat pada Gambar 5.4 yaitu ilustrasi dari sistem *object detection* yang dibuat pada suatu inovasi produk berupa *smartglasses*.

6.2 Saran

Berdasarkan hasil penelitian yang telah didapatkan, maka peneliti dapat memberikan saran sebagai berikut:

1. Mengembangkan sistem dengan menambahkan jumlah *dataset* dan juga ragam atau jenis dari objek pada gambar untuk melatih *model* yang akan dibangun agar dapat mencapai akurasi yang lebih tinggi dan maksimal.
2. Dalam pengumpulan *dataset*, peneliti selanjutnya harus mengumpulkan dan menggunakan gambar dari segala sisi dari setiap objek yang akan digunakan dalam

proses *training*, sehingga sistem yang akan dibangun dapat mengenali objek lebih baik dan mampu menghasilkan tingkat akurasi yang lebih tinggi.

3. Menambahkan jumlah *steps* pada proses *training* sehingga dapat menghasilkan tingkat akurasi yang lebih tinggi dengan menggunakan jumlah *batch* hingga 64 atau 128.
4. Menggunakan spesifikasi perangkat yang lebih tinggi yaitu dengan menggunakan *computer* yang memiliki *Randim Access Memory* (RAM) yang tinggi dan menggunkan *Graphic Processing Unit* (GPU) untuk mempercepat proses *training*.

DAFTAR PUSTAKA

Abdillah, M. F., Nasri, J., & Aditsania, A. (2016). Using Deep Learning to Predict Customer Churn In A Mobile Telecomunication Network. *eProceedings of Engineering*, *3*(2).

Ahmad, W. A. (2022). *Media bantu fungsi kognitif lansia berbasis citra digital dengan yolo/Wildan Ahmad* (Doctoral dissertation, Universitas Negeri Malang).

Argyatiyasa, N., Suprajitno, S., & Martiningsih, W. (2015). Gaya Hidup Sehat Lansia. Jurnal Ners dan Kebidanan (Journal of Ners and Midwifery), 2(3), 222-226.

Ashar, P. H. (2016). Gambaran Persepsi Faktor Risiko Jatuh Pada Lansia Di Panti Sosial Tresna Werdha Budi Mulia 4 Margaguna Jakarta Selatan.

Chatterjee, S., & Roy, S. (2021). A low-cost assistive wheelchair for handicapped & elderly people. *Ain Shams Engineering Journal*, *12*(4), 3835-3841.

Chotivatunyu, P., & Hnoohom, N. (2020, November). Medicine Identification System on Mobile Devices for the Elderly. In *2020 15th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)* (pp. 1-6). IEEE.

Feng, W., Liu, R., & Zhu, M. (2014). Fall detection for elderly person care in a vision-based home surveillance environment using a monocular camera. *signal, image and video processing*, *8*(6), 1129-1138.

Gautam, L. K., & Gulhane, V. S. (2022). Food Assessment *Model* for Indian Elderly Persons Using CNN and Image Processing Techniques. In *Proceedings of Third International Conference on Communication, Computing and Electronics Systems* (pp. 1093-1104). Springer, Singapore.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

Infodatin Kemenkes, R. I. (2016). Situasi Lanjut Usia (Lansia) di Indonesia. *On line at https://pusdatin.kemkes.go.id/resources/download/pusdatin/infodatin/Infodatin-lansia-2016.pdf*

Källstrand-Eriksson, J., Hildingh, C., & Bengtsson, B. (2016). History of falling and visual ability among independently living elderly in Sweden. *Clinical Ophthalmology (Auckland, NZ)*, *10*, 1265.

Kementrian Kesehatan 2012. Profil Kesehatan R.I Jakarta.

Le, Hoang Huy. 2015. Dvision Intro. https://www.academia.edu/27143145/Dvision_intro. Diakses pada 26 Juni 2022.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436-444.

Lu, K. L., & Chu, E. T. H. (2018). An image-based fall detection system for the elderly. *Applied Sciences*, *8*(10), 1995.

Lyu, B., Wang, Z., Li, H., Tanaka, A., Funumoto, K., & Meng, L. (2021). Deep Leaning based Medicine Packaging Information Recognition for Medication Use in the Elderly. *Procedia Computer Science*, *187*, 194-199.

Malida, Dyan. 2011. Faktor Yang Mempengaruhi Tingkat Kemandirian Lansia Dalam Melakukan Aktifitas Kehidupan Sehari – hari Di Panti Sosial Tresna Werdha Budi Luhur Kota Jambi

Marfu, N. J. L. (2020). Perbandingan Antara Svm Dan Cnn Untuk Mendeteksi Objek Kapal Pada Citra Satelit (*Doctoral dissertation*, Universitas Islam Indonesia).

Mukhiddinov, M., & Cho, J. (2021). Smart glass system using deep learning for the blind and visually impaired. Electronics, 10(22), 2756.

Mastura, U. H., Haryotedjo, T., & Widyaevan, D. A. (2016). Perancangan Interior Panti Lansia Di Bandung Berdasarkan Perilaku Lansia. *eProceedings of Art & Design*, *3*(3).

Muslimin, M. (2016). PERAMALAN BEBAN LISTRIK JANGKA MENENGAH PADA SISTEM KELISTRIKAN KOTA SAMARINDA. *Jurnal Ilmiah Teknik Industri*, *14*(2), 113-121.

Naufal, M. F. (2021). Analisis Perbandingan Algoritma Svm, Knn, Dan Cnn untuk Klasifikasi Citra Cuaca. *Jurnal Teknologi Informasi dan Ilmu Komputer*, *8*(2), 311-317.

Nguyen, V. D., Le, M. T., Do, A. D., Duong, H. H., Thai, T. D., & Tran, D. H. (2014, June). An efficient camera-based surveillance for fall detection of elderly people. In *2014 9th IEEE Conference on Industrial Electronics and Applications* (pp. 994-997). IEEE.

Nies, M. A., & McEwen, M. (2007). Senior health. *Community/public health nursing, promoting the health of populations (4th ed., pp. 332–358). Philadelphia: Saunders*.

Nurmila, N., Sugiharto, A., & Sarwoko, E. A. (2010). Algoritma back propagation neural network untuk pengenalan pola karakter huruf jawa. *Jurnal Masyarakat Informatika*, *1*(1), 1-10.

Renaud, J., & Bédard, E. (2013). Depression in the elderly with visual impairment and its association with quality of life. *Clinical interventions in aging*, *8*, 931.

Ruenin, P., Bootkrajang, J., & Chawachat, J. (2020, July). A System to Estimate the Amount and Calories of Food that Elderly People in the Hospital Consume. In *Proceedings of the 11th International Conference on Advances in Information Technology* (pp. 1-7).

Sahasri, M., & Gireesh, C. (2017). Object motion detection and tracking for video surveillance. *International Journal of Engineering Trends and Technology*, 161-164.

Santoso, A., & Gunawan Ariyanto, S. T. (2018). *Implementasi deep learning berbasis keras untuk pengenalan wajah* (Doctoral dissertation, Universitas Muhammadiyah Surakarta).

Yunanto, A. A., & Herumurti, D. (2016, October). Face recognition based on extended symmetric local graph structure. In *2016 International Conference on Information & Communication Technology and Systems (ICTS)* (pp. 80-84). IEEE.

Youm, S., Kim, C., Choi, S., & Kang, Y. S. (2019). Development of a methodology to predict and monitor emergency situations of the elderly based on object detection. *Multimedia Tools and Applications*, *78*(5), 5427-5444.

Zufar, M., & Setiyono, B. (2016). Convolutional neural networks untuk pengenalan wajah secara real-time. Jurnal Sains Dan Seni Its Vol. 5 No. 2 (2016) 2337-3520 (2301-928X Print), A72-A77.

LAMPIRAN

Lampiran 1 Script Konversi Dataset XML to SCV

```
import os
import glob
import pandas as pd
import xml.etree.ElementTree as ET


def xml_to_csv(path):
    xml_list = []
    for xml_file in glob.glob(path + '/*.xml'):
        tree = ET.parse(xml_file)
        root = tree.getroot()
        for member in root.findall('object'):
            value = (root.find('filename').text,
                     int(root.find('size')[0].text),
                     int(root.find('size')[1].text),
                     member[0].text,
                     int(member[4][0].text),
                     int(member[4][1].text),
                     int(member[4][2].text),
                     int(member[4][3].text)
                     )
            xml_list.append(value)
    column_name = ['filename', 'width', 'height', 'class', 'xmin', 'ymin', 'xmax', 'ymax']
    xml_df = pd.DataFrame(xml_list, columns=column_name)
    return xml_df


def main():
    for directory in ('train', 'test'):
        image_path = os.path.join(os.getcwd(), 'images/{}'.format(directory))
        xml_df = xml_to_csv(image_path)
        xml_df.to_csv('data/{}_labels.csv'.format(directory), index=None)
        print('Successfully converted xml to csv.')


main()
```

Lampiran 2 Script Konversi Dataset SCV to TFRecord

```
from __future__ import division
from __future__ import print_function
from __future__ import absolute_import

import os
import io
import pandas as pd
import tensorflow as tf

from PIL import Image
from object_detection.utils import dataset_util
from collections import namedtuple, OrderedDict

flags = tf.compat.v1.flags
flags.DEFINE_string('csv_input', '', 'Path to the CSV input')
flags.DEFINE_string('output_path', '', 'Path to output TFRecord')
flags.DEFINE_string('image_dir', '', 'Path to images')
FLAGS = flags.FLAGS


# TO-DO replace this with label map
def class_text_to_int(row_label):
    if row_label == 'Bantal':
        return 1
    if row_label == 'Bingkai Foto':
        return 2
    if row_label == 'Botol':
        return 3
    if row_label == 'Cotton Buds':
        return 4
    if row_label == 'Galon':
        return 5
    if row_label == 'Gelas':
        return 6
    if row_label == 'Guling':
        return 7
    if row_label == 'Gunting':
        return 8
    if row_label == 'Jam Dinding':
        return 9
    if row_label == 'Kacamata':
        return 10
    if row_label == 'Kalendar':
        return 11
    if row_label == 'Keset Kaki':
        return 12
    if row_label == 'Kipas Angin':
        return 13
    if row_label == 'Koper':
        return 14
    if row_label == 'Korek Gas':
        return 15
    if row_label == 'Kursi':
        return 16
    if row_label == 'Lampu Meja':
        return 17
    if row_label == 'Lemari Pakaian':
        return 18
    if row_label == 'Radio':
        return 19
    if row_label == 'Sapu Lidi':
        return 20
    if row_label == 'Selimut':
        return 21
    if row_label == 'Sisir':
        return 22
    if row_label == 'Spons Mandi':
        return 23
```

```
if row_label == 'Tas':
    return 24
if row_label == 'Termos':
    return 25
else:
    return 0

def split(df, group):

    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]

def create_tf_example(group, path):
    with tf.io.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size

    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []

    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))

    tf_example = tf.train.Example(features=tf.train.Features(feature={
        'image/height': dataset_util.int64_feature(height),
        'image/width': dataset_util.int64_feature(width),
        'image/filename': dataset_util.bytes_feature(filename),
        'image/source_id': dataset_util.bytes_feature(filename),
        'image/encoded': dataset_util.bytes_feature(encoded_jpg),
        'image/format': dataset_util.bytes_feature(image_format),
        'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
        'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
        'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
        'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
        'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
        'image/object/class/label': dataset_util.int64_list_feature(classes),
    }))
    return tf_example


def main(_):
    writer = tf.io.TFRecordWriter(FLAGS.output_path)
    path = os.path.join(FLAGS.image_dir)
    examples = pd.read_csv(FLAGS.csv_input)
    grouped = split(examples, 'filename')
    for group in grouped:
        tf_example = create_tf_example(group, path)
        writer.write(tf_example.SerializeToString())

    writer.close()
    output_path = os.path.join(os.getcwd(), FLAGS.output_path)
    print('Successfully created the TFRecords: {}'.format(output_path))

if __name__ == '__main__':
    tf.compat.v1.app.run()
```

Lampiran 3 Script Konfigurasi Label Map

```
item {
        id 1
        name: 'Bantal'
}
item {
        id 2
        name: 'Bingkai Foto'
}
item {
        id 3
        name: 'Botol'
}
item {
        id 4
        name: 'Cotton Buds'
}
item {
        id 5
        name: 'Galon'
}
item {
        id 6
        name: 'Gelas'
}
item {
        id 7
        name: 'Guling'
}
item {
        id 8
        name: 'Gunting'
}
item {
        id 9
        name: 'Jam Dinding'
}
item {
        id 10
        name: 'Kacamata'
}
item {
        id 11
        name: 'Kalendar'
}
item {
        id 12
        name: 'Keset Kaki'
}
```

```
item {
        id: 13
        name: 'Kipas Angin'
}
item {
        id: 14
        name: 'Koper'
}
item {
        id: 15
        name: 'Korek Gas'
}
item {
        id: 16
        name: 'Kursi'
}
item {
        id: 17
        name: 'Lampu Meja'
}
item {
        id: 18
        name: 'Lemari Pakaian'
}
item {
        id: 19
        name: 'Radio'
}
item {
        id: 20
        name: 'Sapu Lidi'
}
item {
        id: 21
        name: 'Selimut'
}
item {
        id: 22
        name: 'Sisir'
}
item {
        id: 23
        name: 'Spons Mandi'
}
item {
        id: 24
        name: 'Tas'
}
item {
        id: 25
        name: 'Termos'
}
```

Lampiran 4 Script Konfigurasi *Model*

```
model {
  ssd {
    inplace_batchnorm_update: true
    freeze_batchnorm: false
    num_classes: 25
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
        use_matmul_gather: true
      }
    }
    similarity_calculator {
      iou_similarity {
      }
    }
    encode_background_as_zeros: true
    anchor_generator {
      multiscale_anchor_generator {
        min_level: 3
        max_level: 7
        anchor_scale: 4.0
        aspect_ratios: [1.0, 2.0, 0.5]
        scales_per_octave: 2
      }
    }
    image_resizer {
      fixed_shape_resizer {
        height: 640
        width: 640
      }
    }
    box_predictor {
      weight_shared_convolutional_box_predictor {
        depth: 128
        class_prediction_bias_init: -4.6
        conv_hyperparams {
          activation: RELU_6,
          regularizer {
            l2_regularizer {
              weight: 0.00004
            }
          }
          initializer {
            random_normal_initializer {
              stddev: 0.01
              mean: 0.0
            }
          }
          batch_norm {
            scale: true,
            decay: 0.997,
            epsilon: 0.001,
          }
        }
        num_layers_before_predictor: 4
        share_prediction_tower: true
        use_depthwise: true
        kernel_size: 3
      }
    }
```

```
feature_extractor {
      type: 'ssd_mobilenet_v2_fpn_keras'
      use_depthwise: true
      fpn {
        min_level: 3
        max_level: 7
        additional_layer_depth: 128
      }
      min_depth: 16
      depth_multiplier: 1.0
      conv_hyperparams {
        activation: RELU_6,
        regularizer {
          l2_regularizer {
            weight: 0.00004
          }
        }
        initializer {
          random_normal_initializer {
            stddev: 0.01
            mean: 0.0
          }
        }
        batch_norm {
          scale: true,
          decay: 0.997,
          epsilon: 0.001,
        }
      }
      override_base_feature_extractor_hyperparams: true
    }
    loss {
      classification_loss {
        weighted_sigmoid_focal {
          alpha: 0.25
          gamma: 2.0
        }
      }
      localization_loss {
        weighted_smooth_l1 {
        }
      }
      classification_weight: 1.0
      localization_weight: 1.0
    }
    normalize_loss_by_num_matches: true
    normalize_loc_loss_by_codesize: true
    post_processing {
      batch_non_max_suppression {
        score_threshold: 1e-8
        iou_threshold: 0.6
        max_detections_per_class: 100
        max_total_detections: 100
      }
      score_converter: SIGMOID
    }
  }
}

train_config: {
  fine_tune_checkpoint_version: V2
  fine_tune_checkpoint: "/content/gdrive/MyDrive/ObjectDetection/saved_model/ckpt-0"
  fine_tune_checkpoint_type: "detection"
  batch_size: 5
  sync_replicas: true
  startup_delay_steps: 0
  replicas_to_aggregate: 8
  num_steps: 50000
  data_augmentation_options {
    random_horizontal_flip {
    }
  }
```

```
data_augmentation_options {
    random_crop_image {
      min_object_covered: 0.0
      min_aspect_ratio: 0.75
      max_aspect_ratio: 3.0
      min_area: 0.75
      max_area: 1.0
      overlap_thresh: 0.0
    }
  }
  optimizer {
    momentum_optimizer: {
      learning_rate: {
        cosine_decay_learning_rate {
          learning_rate_base: .08
          total_steps: 50000
          warmup_learning_rate: .026666
          warmup_steps: 1000
        }
      }
      momentum_optimizer_value: 0.9
    }
    use_moving_average: false
  }
  max_number_of_boxes: 100
  unpad_groundtruth_tensors: false
}

train_input_reader: {
  label_map_path: "/content/gdrive/MyDrive/ObjectDetection/data/label_map.pbtxt"
  tf_record_input_reader {
    input_path: "/content/gdrive/MyDrive/ObjectDetection/data/train.record"
  }
}

eval_config: {
  metrics_set: "coco_detection_metrics"
  use_moving_averages: false
}

eval_input_reader: {
  label_map_path: "/content/gdrive/MyDrive/ObjectDetection/data/label_map.pbtxt"
  shuffle: false
  num_epochs: 1
  tf_record_input_reader {
    input_path: "/content/gdrive/MyDrive/ObjectDetection/data/test.record"
  }
}
```

Lampiran 5 Script Training *Model*

```
r"""Creates and runs TF2 object detection models.

For local training/evaluation run:
PIPELINE_CONFIG_PATH=path/to/pipeline.config
MODEL_DIR=/tmp/model_outputs
NUM_TRAIN_STEPS=10000
SAMPLE_1_OF_N_EVAL_EXAMPLES=1
python model_main_tf2.py -- \
  --model_dir=$MODEL_DIR --num_train_steps=$NUM_TRAIN_STEPS \
  --sample_1_of_n_eval_examples=$SAMPLE_1_OF_N_EVAL_EXAMPLES \
  --pipeline_config_path=$PIPELINE_CONFIG_PATH \
  --alsologtostderr
"""
from absl import flags
import tensorflow.compat.v2 as tf
from object_detection import model_lib_v2

flags.DEFINE_string('pipeline_config_path', None, 'Path to pipeline config '
                    'file.')
flags.DEFINE_integer('num_train_steps', None, 'Number of train steps.')
flags.DEFINE_bool('eval_on_train_data', False, 'Enable evaluating on train '
                  'data (only supported in distributed training).')
flags.DEFINE_integer('sample_1_of_n_eval_examples', None, 'Will sample one of '
                     'every n eval input examples, where n is provided.')
flags.DEFINE_integer('sample_1_of_n_eval_on_train_examples', 5, 'Will sample '
                     'one of every n train input examples for evaluation, '
                     'where n is provided. This is only used if '
                     '`eval_training_data` is True.')
flags.DEFINE_string(
    'model_dir', None, 'Path to output model directory '
                       'where event and checkpoint files will be written.')
flags.DEFINE_string(
    'checkpoint_dir', None, 'Path to directory holding a checkpoint.  If '
    '`checkpoint_dir` is provided, this binary operates in eval-only mode, '
    'writing resulting metrics to `model_dir`.')

flags.DEFINE_integer('eval_timeout', 3600, 'Number of seconds to wait for an'
                     'evaluation checkpoint before exiting.')

flags.DEFINE_bool('use_tpu', False, 'Whether the job is executing on a TPU.')
flags.DEFINE_string(
    'tpu_name',
    default=None,
    help='Name of the Cloud TPU for Cluster Resolvers.')
flags.DEFINE_integer(
    'num_workers', 1, 'When num_workers > 1, training uses '
    'MultiWorkerMirroredStrategy. When num_workers = 1 it uses '
    'MirroredStrategy.')
```

```
flags.DEFINE_integer(
    'checkpoint_every_n', 1000, 'Integer defining how often we checkpoint.')
flags.DEFINE_boolean('record_summaries', True,
                     ('Whether or not to record summaries defined by the model'
                      ' or the training pipeline. This does not impact the'
                      ' summaries of the loss values which are always'
                      ' recorded.'))

FLAGS = flags.FLAGS


def main(unused_argv):
  flags.mark_flag_as_required('model_dir')
  flags.mark_flag_as_required('pipeline_config_path')
  tf.config.set_soft_device_placement(True)

  if FLAGS.checkpoint_dir:
    model_lib_v2.eval_continuously(
        pipeline_config_path=FLAGS.pipeline_config_path,
        model_dir=FLAGS.model_dir,
        train_steps=FLAGS.num_train_steps,
        sample_1_of_n_eval_examples=FLAGS.sample_1_of_n_eval_examples,
        sample_1_of_n_eval_on_train_examples=(
            FLAGS.sample_1_of_n_eval_on_train_examples),
        checkpoint_dir=FLAGS.checkpoint_dir,
        wait_interval=300, timeout=FLAGS.eval_timeout)
  else:
    if FLAGS.use_tpu:
      # TPU is automatically inferred if tpu_name is None and
      # we are running under cloud ai-platform.
      resolver = tf.distribute.cluster_resolver.TPUClusterResolver(
          FLAGS.tpu_name)
      tf.config.experimental_connect_to_cluster(resolver)
      tf.tpu.experimental.initialize_tpu_system(resolver)
      strategy = tf.distribute.experimental.TPUStrategy(resolver)
    elif FLAGS.num_workers > 1:
      strategy = tf.distribute.experimental.MultiWorkerMirroredStrategy()
    else:
      strategy = tf.compat.v2.distribute.MirroredStrategy()

    with strategy.scope():
      model_lib_v2.train_loop(
          pipeline_config_path=FLAGS.pipeline_config_path,
          model_dir=FLAGS.model_dir,
          train_steps=FLAGS.num_train_steps,
          use_tpu=FLAGS.use_tpu,
          checkpoint_every_n=FLAGS.checkpoint_every_n,
          record_summaries=FLAGS.record_summaries)

if __name__ == '__main__':
  tf.compat.v1.app.run()
```

Lampiran 6 Script Export *Model*

```
r"""Tool to export an object detection model for inference.

Prepares an object detection tensorflow graph for inference using model
configuration and a trained checkpoint. Outputs associated checkpoint files,
a SavedModel, and a copy of the model config.

The inference graph contains one of three input nodes depending on the user
specified option.
  * `image_tensor`: Accepts a uint8 4-D tensor of shape [1, None, None, 3]
  * `float_image_tensor`: Accepts a float32 4-D tensor of shape
    [1, None, None, 3]
  * `encoded_image_string_tensor`: Accepts a 1-D string tensor of shape [None]
    containing encoded PNG or JPEG images. Image resolutions are expected to be
    the same if more than 1 image is provided.
  * `tf_example`: Accepts a 1-D string tensor of shape [None] containing
    serialized TFExample protos. Image resolutions are expected to be the same
    if more than 1 image is provided.
  * `image_and_boxes_tensor`: Accepts a 4-D image tensor of size
    [1, None, None, 3] and a boxes tensor of size [1, None, 4] of normalized
    bounding boxes. To be able to support this option, the model needs
    to implement a predict_masks_from_boxes method. See the documentation
    for DetectionFromImageAndBoxModule for details.

and the following output nodes returned by the model.postprocess(..):
  * `num_detections`: Outputs float32 tensors of the form [batch]
      that specifies the number of valid boxes per image in the batch.
  * `detection_boxes`: Outputs float32 tensors of the form
      [batch, num_boxes, 4] containing detected boxes.
  * `detection_scores`: Outputs float32 tensors of the form
      [batch, num_boxes] containing class scores for the detections.
  * `detection_classes`: Outputs float32 tensors of the form
      [batch, num_boxes] containing classes for the detections.


Example Usage:
--------------
python exporter_main_v2.py \
    --input_type image_tensor \
    --pipeline_config_path path/to/ssd_inception_v2.config \
    --trained_checkpoint_dir path/to/checkpoint \
    --output_directory path/to/exported_model_directory
    --use_side_inputs True/False \
    --side_input_shapes dim_0,dim_1,...dim_a/.../dim_0,dim_1,...,dim_z \
    --side_input_names name_a,name_b,...,name_c \
    --side_input_types type_1,type_2

The expected output would be in the directory
path/to/exported_model_directory (which is created if it does not exist)
holding two subdirectories (corresponding to checkpoint and SavedModel,
respectively) and a copy of the pipeline config.
```

```
Config overrides (see the `config_override` flag) are text protobufs
(also of type pipeline_pb2.TrainEvalPipelineConfig) which are used to override
certain fields in the provided pipeline_config_path.  These are useful for
making small changes to the inference graph that differ from the training or
eval config.

Example Usage (in which we change the second stage post-processing score
threshold to be 0.5):

python exporter_main_v2.py \
    --input_type image_tensor \
    --pipeline_config_path path/to/ssd_inception_v2.config \
    --trained_checkpoint_dir path/to/checkpoint \
    --output_directory path/to/exported_model_directory \
    --config_override " \
            model{ \
              faster_rcnn { \
                second_stage_post_processing { \
                  batch_non_max_suppression { \
                    score_threshold: 0.5 \
                  } \
                } \
              } \
            }"

If side inputs are desired, the following arguments could be appended
(the example below is for Context R-CNN).
   --use_side_inputs True \
   --side_input_shapes 1,2000,2057/1 \
   --side_input_names context_features,valid_context_size \
   --side_input_types tf.float32,tf.int32
"""
from absl import app
from absl import flags

import tensorflow.compat.v2 as tf
from google.protobuf import text_format
from object_detection import exporter_lib_v2
from object_detection.protos import pipeline_pb2

tf.enable_v2_behavior()


FLAGS = flags.FLAGS

flags.DEFINE_string('input_type', 'image_tensor', 'Type of input node. Can be '
                    'one of [`image_tensor`, `encoded_image_string_tensor`, '
                    '`tf_example`, `float_image_tensor`, '
                    '`image_and_boxes_tensor`]')
flags.DEFINE_string('pipeline_config_path', None,
                    'Path to a pipeline_pb2.TrainEvalPipelineConfig config '
                    'file.')
```

```
flags.DEFINE_string('trained_checkpoint_dir', None,
                    'Path to trained checkpoint directory')
flags.DEFINE_string('output_directory', None, 'Path to write outputs.')
flags.DEFINE_string('config_override', '',
                    'pipeline_pb2.TrainEvalPipelineConfig '
                    'text proto to override pipeline_config_path.')
flags.DEFINE_boolean('use_side_inputs', False,
                     'If True, uses side inputs as well as image inputs.')
flags.DEFINE_string('side_input_shapes', '',
                    'If use_side_inputs is True, this explicitly sets '
                    'the shape of the side input tensors to a fixed size. The '
                    'dimensions are to be provided as a comma-separated list '
                    'of integers. A value of -1 can be used for unknown '
                    'dimensions. A `/` denotes a break, starting the shape of '
                    'the next side input tensor. This flag is required if '
                    'using side inputs.')
flags.DEFINE_string('side_input_types', '',
                    'If use_side_inputs is True, this explicitly sets '
                    'the type of the side input tensors. The '
                    'dimensions are to be provided as a comma-separated list '
                    'of types, each of `string`, `integer`, or `float`. '
                    'This flag is required if using side inputs.')
flags.DEFINE_string('side_input_names', '',
                    'If use_side_inputs is True, this explicitly sets '
                    'the names of the side input tensors required by the model '
                    'assuming the names will be a comma-separated list of '
                    'strings. This flag is required if using side inputs.')

flags.mark_flag_as_required('pipeline_config_path')
flags.mark_flag_as_required('trained_checkpoint_dir')
flags.mark_flag_as_required('output_directory')


def main(_):
  pipeline_config = pipeline_pb2.TrainEvalPipelineConfig()
  with tf.io.gfile.GFile(FLAGS.pipeline_config_path, 'r') as f:
    text_format.Merge(f.read(), pipeline_config)
  text_format.Merge(FLAGS.config_override, pipeline_config)
  exporter_lib_v2.export_inference_graph(
      FLAGS.input_type, pipeline_config, FLAGS.trained_checkpoint_dir,
      FLAGS.output_directory, FLAGS.use_side_inputs, FLAGS.side_input_shapes,
      FLAGS.side_input_types, FLAGS.side_input_names)


if __name__ == '__main__':
  app.run(main)
```

Lampiran 7 Script Pengujian *Model*s Pada Google Colaboratory

```
import io
import os
import scipy.misc
import numpy as np
import six
import time
import glob
from IPython.display import display
from six import BytesIO

import matplotlib
import matplotlib.pyplot as plt
from PIL import Image, ImageDraw, ImageFont

import tensorflow as tf
from object_detection.utils import ops as utils_ops
from object_detection.utils import label_map_util
from object_detection.utils import visualization_utils as vis_util
%matplotlib inline
category_index = label_map_util.create_category_index_from_labelmap('/content/gdrive/MyDrive/
                                ObjectDetection/data/label_map.pbtxt', use_display_name=True)

tf.keras.backend.clear_session()
model = tf.saved_model.load(f'//content/gdrive/MyDrive/ObjectDetection/inference_graph/saved_model')

def run_inference_for_single_image(model, image):
  image = np.asarray(image)
  # The input needs to be a tensor, convert it using `tf.convert_to_tensor`.
  input_tensor = tf.convert_to_tensor(image)
  # The model expects a batch of images, so add an axis with `tf.newaxis`.
  input_tensor = input_tensor[tf.newaxis,...]
  # Run inference
  model_fn = model.signatures['serving_default']
  output_dict = model_fn(input_tensor)
  # All outputs are batches tensors.
  # Convert to numpy arrays, and take index [0] to remove the batch dimension.
  # We're only interested in the first num_detections.
  num_detections = int(output_dict.pop('num_detections'))
  output_dict = {key:value[0, :num_detections].numpy()
                 for key,value in output_dict.items()}
  output_dict['num_detections'] = num_detections
  # detection_classes should be ints.
  output_dict['detection_classes'] = output_dict['detection_classes'].astype(np.int64)
  # Handle models with masks:
  if 'detection_masks' in output_dict:
    # Reframe the the bbox mask to the image size.
    detection_masks_reframed = utils_ops.reframe_box_masks_to_image_masks(
              output_dict['detection_masks'], output_dict['detection_boxes'],
               image.shape[0], image.shape[1])
    detection_masks_reframed = tf.cast(detection_masks_reframed > 0.5,
                                       tf.uint8)
    output_dict['detection_masks_reframed'] = detection_masks_reframed.numpy()

  return output_dict
```

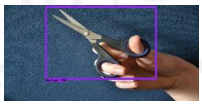
```
!mkdir test_images

for image_path in glob.glob('/content/gdrive/MyDrive/ObjectDetection/data/images/Test_Images/*.jpg'):
  image_np = load_image_into_numpy_array(image_path)
  output_dict = run_inference_for_single_image(model, image_np)
  vis_util.visualize_boxes_and_labels_on_image_array(
      image_np,
      output_dict['detection_boxes'],
      output_dict['detection_classes'],
      output_dict['detection_scores'],
      category_index,
      instance_masks=output_dict.get('detection_masks_reframed', None),
      use_normalized_coordinates=True,
      line_thickness=8)
  display(Image.fromarray(image_np))
```

Lampiran 8 Hasil Pengujian *Model*

Nama Produk	Hasil Deteksi	Tingkat Akurasi	Rata-Rata Akurasi
Bantal		82%	90%
		95%	
		94%	
		89%	
		92%	
Guling		87%	91%
		88%	
		91%	
		93%	
		96%	
Sisir		91%	92%
		96%	
		96%	
		92%	
		84%	

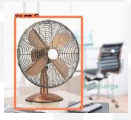
Jam Dinding		98%	98%
		98%	
		97%	
		98%	
		99%	
Lemari Pakaian		99%	95%
		91%	
		92%	
		97%	
		98%	
Kursi		95%	96%
		95%	
		96%	
		96%	
		96%	

Koper		98%	93%
		95%	
		96%	
		94%	
		83%	
Radio		95%	96%
		94%	
		95%	
		98%	
		96%	
Selimut		97%	94%
		92%	
		90%	
		95%	
		94%	

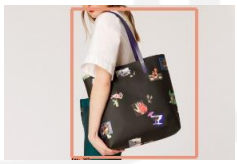
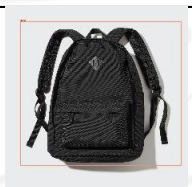
Kacamata		92%	94%
		97%	
		93%	
		94%	
		92%	
Bingkai		98%	94%
		92%	
		95%	
		90%	
		93%	
Gunting		98%	96%
		94%	
		96%	
		96%	
		95%	

Botol		97%	95%
		95%	
		96%	
		97%	
		92%	
Lampu Meja		97%	95%
		95%	
		97%	
		92%	
		95%	
Spons Mandi		96%	95%
		97%	
		96%	
		91%	
		95%	

Galon		89%	93%
		97%	
		92%	
		97%	
		91%	
Sapu Lidi		94%	95%
		92%	
		95%	
		98%	
		97%	
Termos		97%	98%
		99%	
		96%	
		98%	
		98%	

Keset Kaki		97%	96%
		95%	
		95%	
		95%	
		97%	
Kipas Angin		97%	97%
		98%	
		97%	
		96%	
		95%	
Korek Gas		94%	95%
		92%	
		97%	
		95%	
		96%	

Cotton Buds		94%	96%
		95%	
		97%	
		97%	
		97%	
Kalendar		96%	95%
		97%	
		96%	
		94%	
		92%	

Gelas		98%	96%
		97%	
		98%	
		97%	
		92%	
Tas		98%	96%
		93%	
		97%	
		97%	
		94%	
Rata-Rata Akurasi Keseluruhan			95%