Journal of Artificial Intelligence and Software Engineering
Vol. 5, No. 1, Maret 2025, pp. 103-115
E-ISSN: 2777-001X, DOI: 10.30811/jaise.v5i1.6419

The Object and Distance Detection for the Visually Impaired Using Deep Learning ResNet-152 and the Triangulation Method

Muhammad Ichwan[1], Irma Amelia Dewi[2*], Nadiati Salsabilla [3]
[1,2,3] Program Studi Informatika, Fakultas Teknologi Industri, Institut Teknologi Nasional Bandung, Bandung, 40124, Indonesia

Informasi Artikel

Diterima : 10 Februari 2025
Revisi : 16 Februari 2025
Publikasi : 20 Maret 2025

Kata Kunci:

Object Detection
Convolution Neural Network
RetinaNet
ResNet-152
Triangulation
Tunanetra

ABSTRAK

Penelitian ini bertujuan untuk mendeteksi objek dan menentukan jarak dari mobile kamera untuk memudahkan dan membantu pengguna tunanetra dalam mengenali lingkungan sekitar menggunakan beberapa model yang dibuat dengan Metode RetinaNet dan arsitektur Residual Network-152. Terdapat tiga model pendeteksian objek yang dihasilkan dari parameter optimaizer ADAM dan SGD. Pengenalan objek dilakukan menggunakan framework tensorflow dengan dataset sebanyak 2444 gambar. Model pertama dengan parameter training yang digunakan yaitu optimaizer ADAM, epoch 50, batchsize 16, dan lr 1e-5. Model kedua dengan parameter training optimaizer ADAM, epoch 100, batchsize 16, dan lr 1e-5. Model ketiga menggunakan parameter training optimaizer SGD, epoch 50, batchsize 16, dan lr 1e-5. Berdasarkan 250 kali pengujian pada masing-masing model, diperoleh hasil bahwa model terbaik adalah model pertama yang menunjukan nilai precision sebesar 82%, nilai recall sebesar 98%, nilai f1 score sebesar 89% dan nilai accuracy sebesar 86%. Pengujian jarak dari mobile kamera diujikan dalam kelipatan 10 di jarak 100-300 cm dengan tinggi kamera 100-130 cm dan angle kamera 80º-90º mendapatkan hasil deteksi jarak yang baik di tinggi kamera 130 cm karena mendapatkan nilai total selisih paling kecil yaitu 14.3 cm.

ABSTRACT

This research aims to detect objects and determine the distance from the mobile camera to facilitate and assist visually impaired users in recognizing the surrounding environment using several models made with the RetinaNet Method and Residual Network-152 architecture. Three object detection models were generated by ADAM and SGD parameter optimizers. Object recognition was performed using the TensorFlow framework with a dataset of 2,444 images. The first model with training parameters used is ADAM optimizer, epoch 50, batch size 16, and lr 1e-5. The second model has training parameters, such as ADAM optimizer, epoch 100, batch size 16, and lr 1e-5. The third model uses SGD optimizer training parameters, epoch 50, batch size 16, and lr 1e-5. Based on 250 tests on each model, the results show that the best model is the first model, which shows a precision value of 82%, a recall value of 98%, an f1 score value of 89%, and an accuracy value of 86%. The distance from the mobile camera is tested in multiples of 10 at a distance of 100-300 cm with a camera height of 100-130 cm and a camera angle of 80º-90º getting reasonable distance detection results at a camera height of 130 cm because it gets the smallest total difference value of 14.3 cm.

***Penulis Koresponden**

Email: irma_amelia@itenas.ac.id

Cara sitasi IEEE:

M. Ichwan, I.A. Dewi & N. Salsabila, "The Object and Distance Detection for the Visually Impaired Using Deep Learning ResNet-152 and the Triangulation Method" *Journal of Artificial Intelligence and Software Engineering (J-AISE)*, vol. 5, no. 1, 103-115, Maret 2025. doi: 10.30811/jaise.v5i1.6419

1. PENDAHULUAN

Menurut studi yang dilakukan pada tahun 2015 menunjukan sekitar 36 juta jiwa menderita kebutaan, 217 juta jiwa penderita gangguan penglihatan, dan 188 juta jiwa memiliki gangguan penglihatan ringan. Data juga menunjukan bahwa 1.09 miliar orang berusia 35 tahun keatas terkena gangguan penglihatan dekat[1]. Dengan kekurangan fisik yang dialami oleh tunanetra membuat mereka tidak dapat mengetahui secara pasti objek yang ada di sekitarnya. Begitu pula dengan jarak dari objek tersebut, penyandang tunanetra tidak dapat mengetahui jarak dari suatu benda yang ada di depannya.

Penelitian ini dilakukan untuk menentukan objek dan jarak dimana dijelaskan dalam penelitian pada tahun 2017 tentang deteksi objek dan jarak menggunakan sensor ultrasonik berbasis fuzzy yang dilakukan oleh elisawati menjelaskan bahwa pengujian dilakukan dengan menjalankan alat deteksi objek dengan mengukur jarak output yang dihasilkan dan jarak output tersebut dibandingkan dengan data aktual yang sebenarnya dimana data output akan ditampilkan di IC ISD dengan output suara depan dekat, depan sedang, dan depan jauh. Diketahui error dari output sensor ultrasonik hanya 0.05% [2]. Namun penelitian ini tidak menampilkan jarak benda dalam satuan angka. Penelitian dengan topik yang sama juga dilakukan oleh rama okta wiyagi dan muhammad yusvin mustar pada tahun 2015. Penelitian ini dilakukan secara real time menggunakan kamera tunggal pada objek bercahaya. Pengukuran jarak dilakukan dengan dengan memanfaatkan luasan sebuah objek bercahaya yang tertangkap kamera yang tertangkap kamera didapatkan informasi jarak antara objek tersebut dengan kamera, pengujian yang dilakukan mendapat error sebesar 5.51% [3]. Penelitian ini hanya bisa mendeteksi objek yang bercahaya saja sehingga objek yang tidak memiliki cahaya tidak dapat dideteksi oleh sistem. Selanjutnya penelitian tentang deteksi objek secara realtime kembali dilakukan pada tahun 2020 dengan menggunakan metode haar cascade classifier yang dilakukan oleh muhammad rizky pratama, dkk. Penelitian ini menjelaskan tentang sistem deteksi objek secara realtime dimana pengujian dilakukan dengan kamera cctv yang memiliki akurasi sebesar 63.25% untuk deteksi objek. Penelitian ini mendapatkan kesimpulan bahwa, jarak sangat berpengaruh saat pendeteksian objek. Semakin dekat objek dari kamera cctv maka tingkat pendeteksian semakin bagus. Kinerja kamera cctv juga menjadi lebih lambat saat pengambilan video realtime karena diberi algoritma sistem deteksi objek [4].

Untuk proses deteksi jarak dilakukan penelitian pada tahun 2019 [5] oleh Rais yang menjelaskan tentang deteksi jarak menggunakan stereo vision dengan CNN di dalamnya. Namun keakurasian dalam mendeteksi jarak masih memiliki persentase error yang tinggi sekitar 6.1% dengan posisi objek harus berada ditengah antara dua kamera. Penelitian tentang CNN untuk deteksi objek juga dilakukan oleh Rahul, B.Nair pada tahun 2018 menggunakan arsitektur MobileNet [6] yang mendapatkan akurasi deteksi objek sebesar 84%. Deteksi objek dilakukan menggunakan metode 3D Point Cloud Construction. Kemudian untuk deteksi jarak dilakukan penelitian tentang pendeteksian jarak antar kendaraan di jalan raya pada tahun 2014 yang dilakukan oleh Arenado, dkk [7] dimana deteksi objek dilakukan pada plat nomor kendaraan yang kemudian dihitung jaraknya menggunakan radar. Penelitian dengan objek kendaraan juga dilakukan oleh ali pada tahun 2016 [8] menelitian ini membahas tentang estimasi jarak dan deteksi posisi kendaraan dengan menggunakan monocular camera. Estimasi jarak dilakukan pada kamera binocular berupa kamera onboard pada kendaraan dengan menggunakan metode hough transform dan filter kalman sedangkan untuk mendeteksi posisi kendaraan penelitian ini menggunakan metode sistem Lane Departure Warning. Selanjutnya pada tahun 2017 kembali dilakukan penelitian deteksi kendaraan dan estimasi jarak antar kendaraan oleh Deng-Yuan Huang, dkk [9] dengan menggunakan lensa tungal. Fitur HOG dan SVM dijadikan sebagai metode untuk mendeteksi objek kendaraan.

Fokus penelitian ini yaitu object detection. Objek detection atau pendeteksian objek menjadi salah satu bidang dalam computer vision dan astificial intelligence (AI) yang paling menarik. Pendeteksian objek merupakan teknologi komputer yang berkaitan dengan computer vision dan image processing yang berhubungan dengan mendeteksi suatu objek dalam citra digital yang dapat berupa warna dan bentuk objek[10]. Terdapat beberapa metode dalam mendeteksi dan mengenali objek pada sebuah gambar, salah satunya adalah Convolutional Neural Network (CNN) yang sering digunakan pada data image.

Perkembangan arsitektur CNN dipengaruhi oleh kompetisi ILSVRC (ImageNet Large Scale Visual Recognition Competition) yang diselenggarakan setiap tahunnya oleh ImageNet. Pada kompetisi tersebut dilakukan evaluasi terhadap tingkat error dari suatu algoritma CNN dalam melakukan deteksi objek dan klasifikasi citra dalam skala besar. Pemenang kompetisi pada tahun 2012 dengan arsitektur CNN bernama AlexNet yang memiliki tingkat error sebesar 16,4%. Pada tahun 2013 pemenang dengan kode clarifai mengurangi tingkat error hingga 11.7%. Kemudian pada tahun 2014 terdapat 2 pemenang yaitu arsitektur VGG dengan jumlah layer sebanyak 19 layer dan nilai error hingga 7,3%. Pemenang kedua yaitu arsitektur GoogleNet dengan 22 layer yang berhasil menurunkan nilai error hingga 6,7 %. [11]. Pada tahun 2015 yang menjadi pemenang dalam kompetisi ini adalah arsitektur ResNet dari Microsoft yang meningkatkan penggunaan layer hingga 152 layer yang berhasil menurunkan nilai error hingga 3,57%.[12] Meskipun mendapatkan nilai error 3,57% akan tetapi layer yang digunakan pada arsitektur ResNet sebanyak 152 layer, sehingga dipilih arsitektur ResNet sebagai backbone dari RetinaNet. Pada table 1 menampilkan pemetaan dari top-5 error rate dari kompetidi ILSVRC.

Tabel. 1 Pemetaan top-5 error rate dari kompetisi ILSVRC

Year	CNN	Place	Top-5 error rate
2012	AlexNet(7)	1st	15.30%
2013	ZFNet()	1st	14.80%
2014	GoogLeNet(19)	1st	6.67%
2014	VGG Net(16)	2st	7.30%
2015	ResNet(152)	1st	3.60%

Berdasarkan hal tersebut penelitian ini dilakukan dengan memanfaatkan algoritma CNN dengan model RetinaNet dan backbone ResNet-152 untuk melakukan deteksi objek. Kemudian penelitian untuk deteksi jarak dilakukan untuk mendapatkan akurasi yang lebih baik dari penelitian sebelumnya.

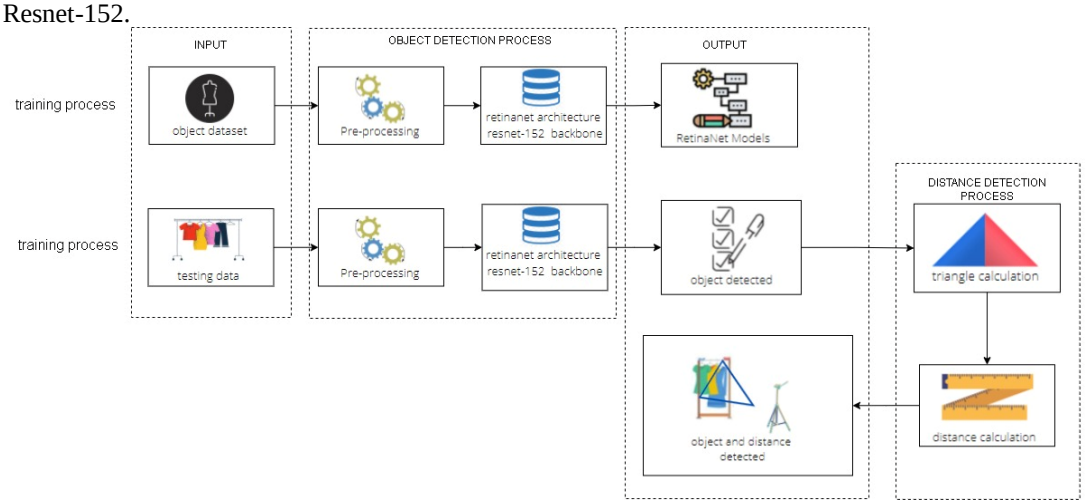
2. METODE

Pada blok diagram sistem deteksi objek terdapat 2 proses yaitu proses training dan testing. Pada proses training. Pada proses training, dilakukan pembuatan model RetinaNet dengan backbone arsitektur ResNet-152. Dataset berupa image dilakukan pembuatan bounding box yang berisi koordinat objek dan label kelas objek yang disimpan pada file dengan ekstensi *.xml. Hasil akhir dari proses training yaitu model RetinaNet dengan arsitektur ResNet-152 yang disimpan dalam file dengan ekstensi *.h5.

Pada proses testing, yang menjadi input dari sistem yaitu citra objek. Citra objek kemudian dilakukan proses preprocessing untuk membuat zero padding pada setiap channel warna dengan cara mengurangi matriks citra BGR terhadap filter mode caffe. Filter mode caffe dilakukan dengan rumus pada Persamaan (1).

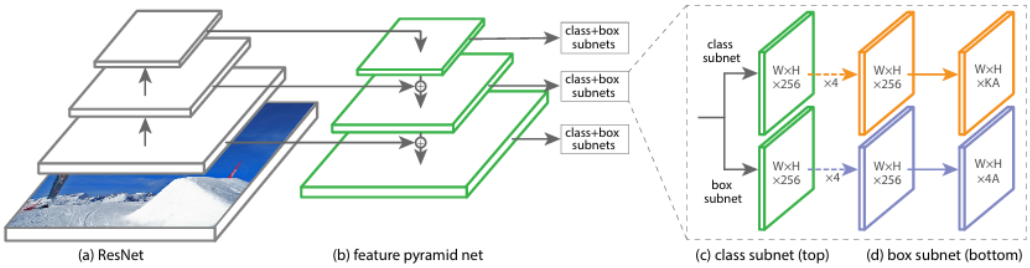
B […, 0] – 103,939
G […, 1] – 116,779 (1)
R […, 2] – 123, 68

Setelah filter mode caffe kemudian dilakukan ekstraksi feature map dengan model RetinaNet dengan arsitektur Resnet-152.



Gambar. 1 Blok diagram sistem deteksi objek

RetinaNet adalah jaringan tunggal atau terpadu yang terdiri dari satu jaringan backbone dan dua subnetwork. Jaringan backbone memiliki tanggung jawab untuk menghitung feature map secara konvolusional pada seluruh citra. Pada subnetwork pertama memiliki tugas untuk melakukan klasifikasi pada objek hasil konvolusi dari jaringan backbone, kemudian pada subnetwork kedua bertugas untuk membuat regresi bounding box. [13].

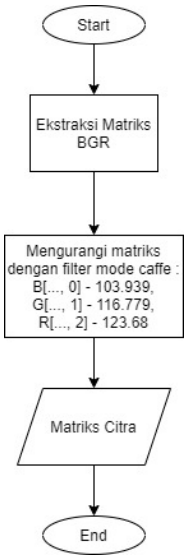


Gambar. 2 Arsitektur RetinaNet [13]

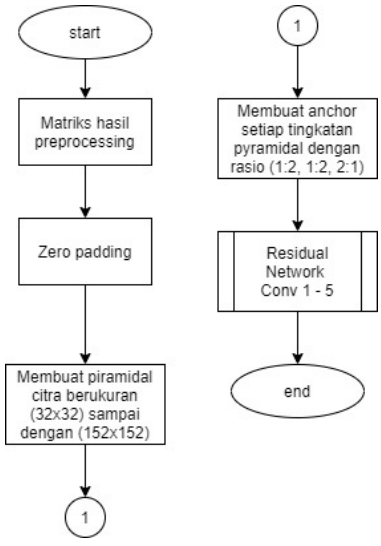
Setelah proses feature map, dilakukan proses anchor box. Setiap anchor bertugas untuk memprediksi adanya objek dan setiap anchor dilakukan proses konvolusi dengan backbone arsitektur ResNet-152 sehingga menghasilkan nilai feature map/bobot untuk memprediksi adanya objek yang dikenali berdasarkan dataset. Selanjutnya dilakukan proses regresi box dan proses klasifikasi. Proses regresi box berguna untuk meregresi setiap kelebihan nilai pada bounding box objek yang terdeteksi. Proses klasifikasi berguna untuk mendeteksi adanya objek sehingga menghasilkan label dan kelas objek.

2.1 Proses Preprocessing

Proses preprocessing mengekstraksi matriks citra BGR. Citra input dalam format RGB (Reg, Green Blue) diubah menjadi matrik yang merepresentasikan nilai pixel dari setiap channel warna BGR (Blue, Green, Red). Matriks citra BGR yang berhasil diekstraksi dilakukan perngurangan dengan filter mode Caffe seperti pada Persamaan (1). Filter mode Caffe ini merupakan teknik untuk memproses citra dengan mengurangi nilai intensitas pixel setiap koordinat citra asli dengan nilai rata-rata atau mean dari dataset. Hal ini bertujuan untuk menormalisasikan citra agar mengurangi variasi yang tidak diperlukan dan fokus pada fitur yang penting dalam citra sehingga dapat diproses pada tahap lebih lanjut.



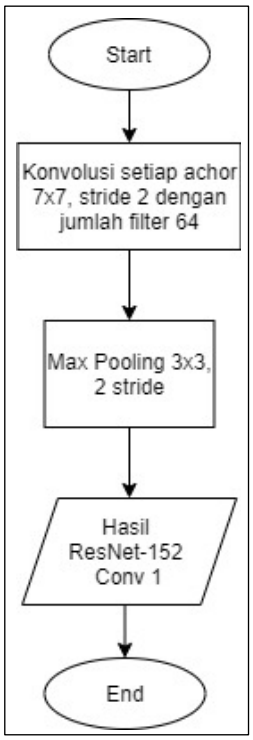
Gambar.3 flowchart preprocessing



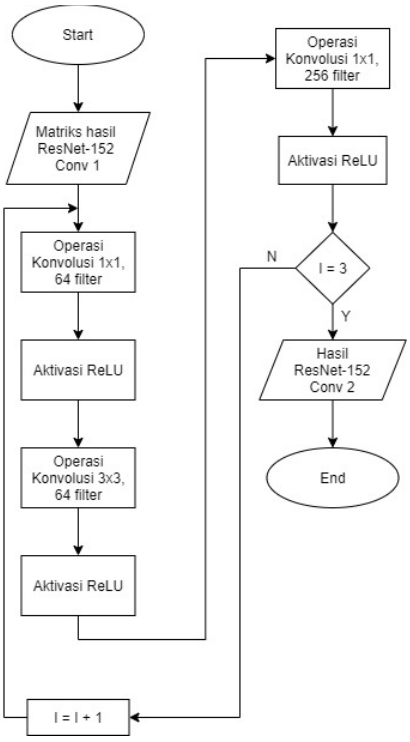
Gambar. 4 Proses ekstraksi feature map

2.2 Proses Ekstraksi Feature Map

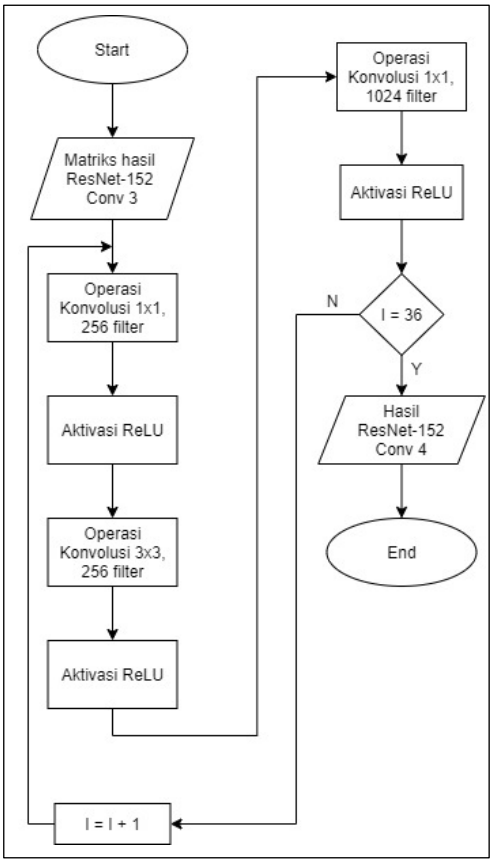
Proses ini dilakukan untuk menghitung feature map secara konvolusional pada seluruh citra seperti Gambar 4 yang menjelaskan tentang alur kerja ekstraksi feature map model RetinaNet menggunakan arsitektur ResNet-152. Proses preprocessing menghasilkan matriks yang kemudian dilakukan proses zero padding dimana proses ini menambahkan dimensi matriks pada sisi citra dengan angka 0, sehingga dimensi matriks citra menjadi lebih besar. Selanjutnya subproses residual network, pada proses ResNet yang dilakukan menggunakan 152 layer dengan proses convolution 1 sampai convolution 5. Proses yang dilakukan terdiri dari konvolusi 7x7, max pooling 3x3, konvolusi 1x1, aktivasi ReLU, operasi konvolusi 3x3, aktivasi ReLU, operasi konvolusi 1x1, jumlah filter yang digunakan pada setiap operasi konvolusi pada modul residual adalah 152 karena disesuaikan dengan tingkatan layer residual network yang digunakan. Proses ResNet menghasilkan nilai feature map/bobot untuk memprediksi adanya objek berdasarkan model dataset yang telah dibuat. Pada Gambar 5 sampai dengan Gambar 9 merupakan *flowchart* dari *subproses residual network conv* 1-5.



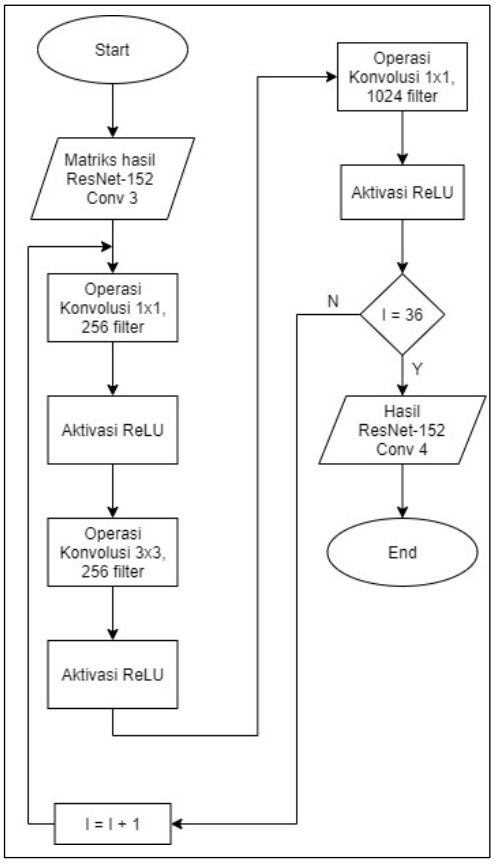
Gambar. 5 Proses ResNet 152 Convolution 1



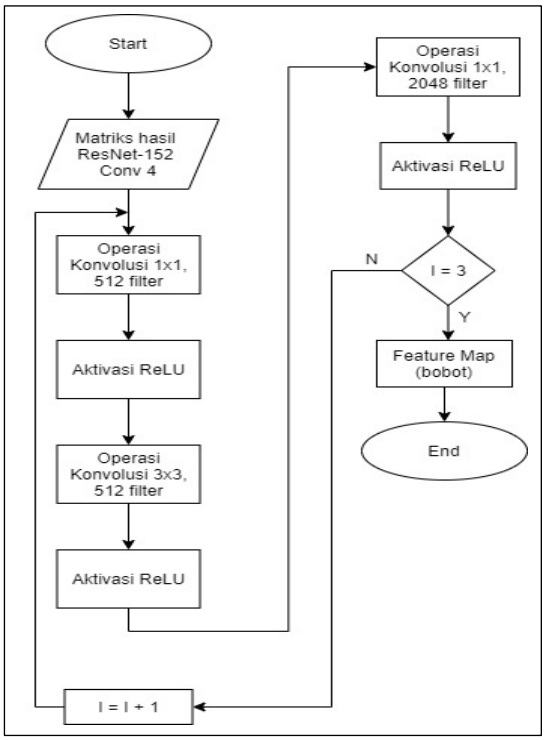
Gambar. 5 Proses ResNet 152 Convolution 2



Gambar. 6 Proses ResNet 152 Convolution 3

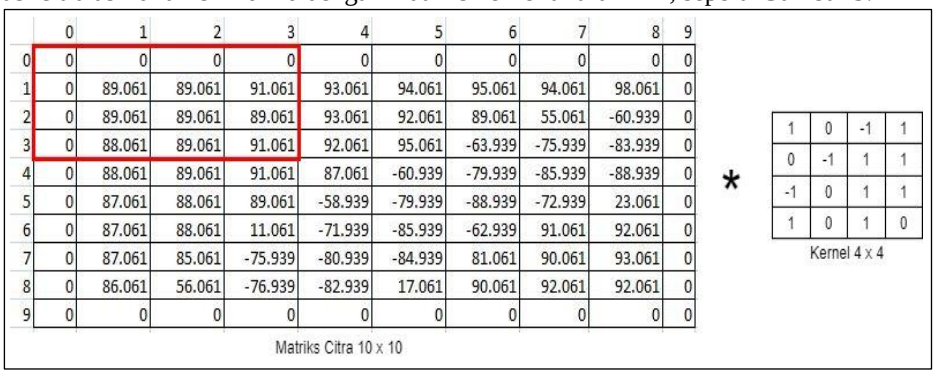


Gambar. 7 Proses ResNet 152 Convolution 4



Gambar. 8 Proses ResNet 152 Convolution 5

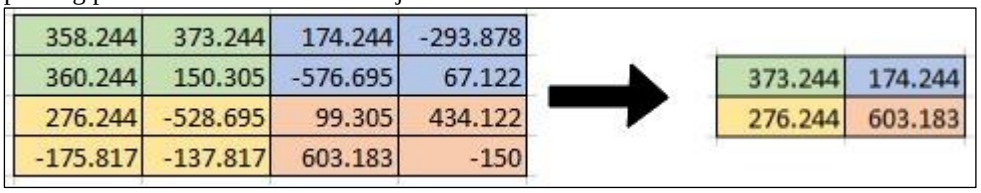
Tahap awal proses *residual network* yaitu mengambil nilai hasil zero padding. Tahap selanjutnya yaitu melakukan proses konvolusi dengan matriks filter 7x7 dengan pergeseran pixel sebanyak dua stride. Proses konvolusi dilakukan dengan cara mengalikan nilai pixel citra dengan nilai pixel kernel citra, sebagai contoh proses konvolusi citra asli di ambil 10x10 dengan matriks kernel ukuran 4x4, seperti Gambar 9.



	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	89.061	89.061	91.061	93.061	94.061	95.061	94.061	98.061	0
2	0	89.061	89.061	89.061	93.061	92.061	89.061	55.061	-60.939	0
3	0	88.061	89.061	91.061	92.061	95.061	-63.939	-75.939	-83.939	0
4	0	88.061	89.061	91.061	87.061	-60.939	-79.939	-85.939	-88.939	0
5	0	87.061	88.061	89.061	-58.939	-79.939	-88.939	-72.939	23.061	0
6	0	87.061	88.061	11.061	-71.939	-85.939	-62.939	91.061	92.061	0
7	0	87.061	85.061	-75.939	-80.939	-84.939	81.061	90.061	93.061	0
8	0	86.061	56.061	-76.939	-82.939	17.061	90.061	92.061	92.061	0
9	0	0	0	0	0	0	0	0	0	0

Gambar. 9 Ilustrasi proses perhitungan konvolusi matriks channel B 10x10 dengan filter 4x4

Setelah dilakukan proses konvolusi selanjutnya akan dilakukan proses max pooling dengan matriks kernel ukuran 3x3 dengan pergeseran pixel sebanyak 2 stride, sebagai contoh matriks hasil proses konvolusi dengan ukuran 4x4 seperti pada Gambar 10, dilakukan proses max pooling dengan ukuran matriks kernel 2x2 dengan jumlah pergeseran pixel sebanyak 2 stride sehingga menghasilkan nilai maximal dengan ukuran matriks 2x2 pada citra channel B, hal ini dilakukan untuk mereduksi ukuran matriks. Pada Gambar 10 merupakan hasil dari proses max pooling pada matriks channel B menjadi 2x2.



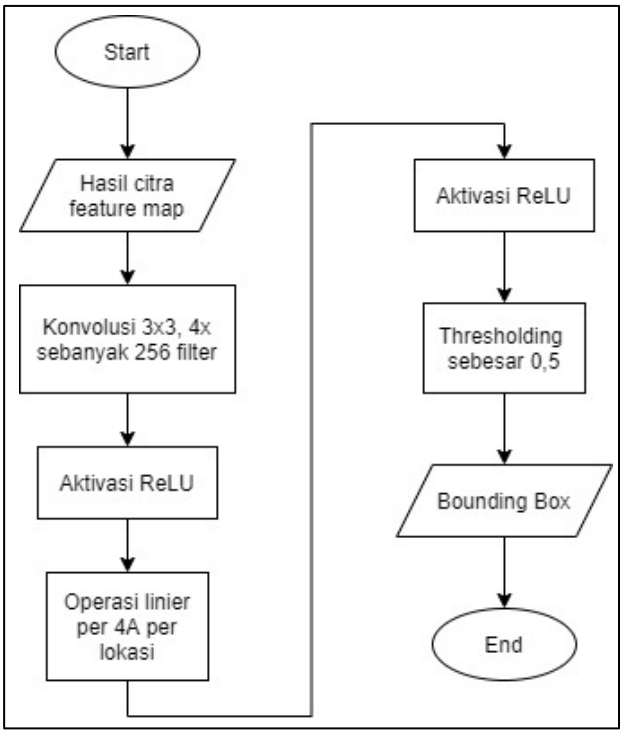
358.244	373.244	174.244	-293.878
360.244	150.305	-576.695	67.122
276.244	-528.695	99.305	434.122
-175.817	-137.817	603.183	-150

373.244	174.244
276.244	603.183

Gambar. 10 Ilustrasi proses Max Pooling citra matriks channel B

2.3 Proses Regresi Box

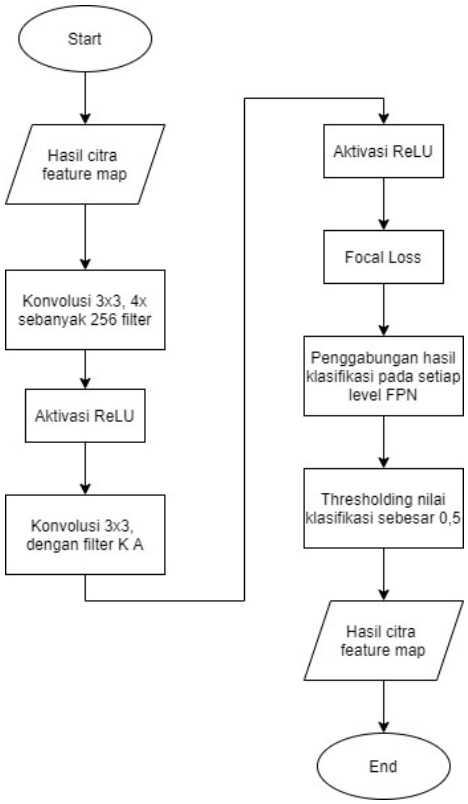
Proses regresi box berguna untuk meregresi kelebihan nilai pada bounding box objek yang terdeteksi. Pada Gambar 11 digambarkan alur kerja dari subproses regresi box. Pada proses regresi, tahap awal yang dilakukan yaitu mengambil nilai hasil citra feature map. Tahap kedua yaitu melakukan konvolusi 3x3 sebanyak 3 kali dengan 256 filter. Tahap ketiga dilakukan aktivasi ReLU untuk mengubah nilai negative menjadi nilai 0. Tahap selanjutnya dilakukan operasi 4 (A=anchor) per lokasi spasial, untuk setiap 4 anchor A per lokasi spasial ini dapat memprediksi *relative offset* diantara *anchor* dan *ground-trurth box.* Tahap kelima aktivasi ReLU dilakukan kembali. Tahap keenam dilakukan fungsi non maximum suppression dengan cara setiap anchor yang di prediksi objek akan dilakukan thresholding score 0,5. Jika *confidence score* < 0,5 maka *bounding box* akan dihapus, jika *confidence score* > 0,5 maka menghasilkan satu *bounding box* yang memprediksi objek dengan nilai score tertinggi. *Bounding box* dengan nilai score tertinggi akan menampilkan objek yang dikenali berdasarkan dataset yang telah dibuat. *Confident score* didapat dari hasil setiap anchor yang diprediksi objek.



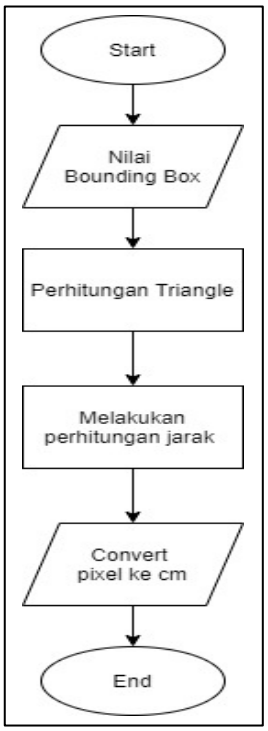
Gambar. 11 Proses Regresi Box

2.4 Proses Klasifikasi

Proses klasifikasi dilakukan untuk mendeteksi adanya objek atau tidak. Proses ini dilakukan bersamaan dengan proses regresi box sehingga menghasilkan nilai, bounding box dan label terhadap objek yang dideteksi. Pada Gambar 12 digambarkan alur kerja subproses klasifikasi. Tahap awal dari proses klasifikasi yaitu mengambil nilai hasil citra dari proses feature map. Kemudian tahapan kedua dilakukan proses konvolusi 3x3 sebanyak 4x dengan jumlah filter sebanyak 256. Tahap ketiga yaitu dilakukan proses aktifasi ReLU untuk mengubah nilai negative menjadi nilai 0. Tahap keempat yaitu dilakukan kembali konvolusi 3x3 dengan filter K = kelas dan A = anchor. Tahapan selanjutnya yaitu dilakukan proses aktivasi ReLU. Selanjutnya dilakukan proses perhitungan focal loss untuk menghilangkan nilai loss pada ground truth label kelas yang terdeteksi. Dalam melakukan proses focal loss menggunakan parameter nilai γ yang digunakan sebesar 2 dan nilai α yang digunakan sebesar 0, 25 untuk mendapatkan hasil maksimal pada penggunaan fungsi focal loss [13]. Tahapan terakhir adalah penggabungan seluruh hasil prediksi tersebut dan dilakukan thresholding sebesar 0, 5.



Gambar. 12 Proses Klasifikasi



Gambar. 13 Proses Deteksi Jarak

2.5 Proses Deteksi Jarak

Proses deteksi objek disajikan dalam flowchart pada gambar 18 berikut. Hasil dari proses deteksi objek berupa nilai bounding box akan menjadi inputan dalam proses deteksi jarak. Setelah objek terdeteksi sistem akan masuk ke proses deteksi jarak dimana sistem akan memulai perhitungan triangle/segitiga. Untuk mendapatkan nilai angle dari segitiga terdapat beberapa nilai yang harus ditentukan, pertama menentukan nilai titik titik koordinat dari bounding box nilai yang diperlukan untuk mendapatkan titik-titik koordinat yaitu nilai titik tengah dari koordinat x (x_center),y (y_center) serta nilai lebar (width_box) dan tinggi (height_box) dari bounding box. Nilai dari parameter ini didapat dari nilai ujung bounding box yaitu b[0]=121, b[1]=0, b[2]=495, b[3]=479 dengan nilai width objek yaitu 640 dan nilai height objek 480. Selanjutnya mencari nilai tengah dari titik x dan y dengan perhitungan Persamaan (2)-Persamaan(5):

$$x\_center = b[0] * width\ objek \quad (2)$$

$$Y\_center = b[1] * height\ objek \quad (3)$$

$$width\_box = b[2] * width\ objek \quad (4)$$

$$height\_box = b[3] * height\ objek \quad (5)$$

Setelah nilai tengah dari koordinat x,y serta nilai tinggi dan lebar dari bounding box diketahui maka nilai titik-titik koordinat dari bounding box dapat diketahui dengan perhitungan Persamaan (6)- Persamaan (9):

$$x1 = x\_center + (width\_box * 0.5) \quad (6)$$

$$y1 = y\_center + (height\_box * 0.5) \quad (7)$$

$$x2 = x\_center - (width\_box * 0.5) \quad (8)$$

$$y2 = y\_center - (height\_box * 0.5) \quad (9)$$

Setelah nilai dari titik-titik koordinat bounding box diketahui langkah selanjutnya menentukan nilai garis pada sisi kiri segitiga, sisi bawah segitiga dan sisi kanan segitiga dengan perhitungan Persamaan (10)-Persamaan(12)

$$Sisi\ segitiga\ bagian\ kiri = x1\ ,\ y2 - (height\_box/7) \quad (10)$$

$$Sisi\ bawah\ segitiga = Width\ objek\ /2\ ,\ 0.9 * height\ objek \quad (11)$$

$$Sisi\ kanan\ segitiga = x2\ ,\ y2 - (height\_box/7) \quad (12)$$

Sekarang sistem mengetahui 3 garis segitiga, selanjutnya menentukan sudut antara titik bawah dan titik kanan dari segitiga dengan menggunakan rumus degrees dan atan2 sehingga nilai dari sudut antara titik bawah dan titik kanan segitiga adalah 30 dan nilai sudut diantara kedua titik tersebut adalah -56. Kedua nilai ini kemudian ditentukan titik angle kanan dan kiri dari segitiga dengan dilakukan penambahan dan pengurangan terhadap 90 dengan Persamaan(13)-(14)

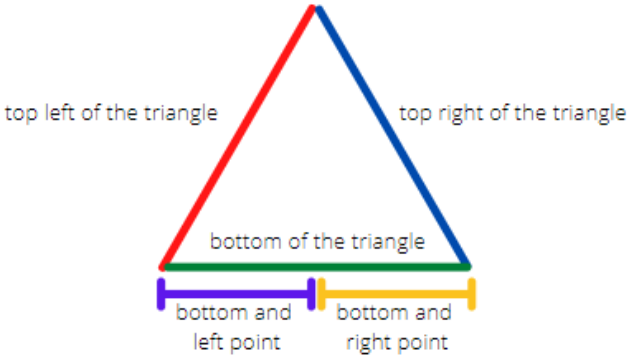
$$\text{Angle kanan} = 90 + 30 = 120 \quad (13)$$

$$\text{Angle kiri} = 90 - (-56) = 146 \quad (14)$$

Kedua nilai angle ini kemudian ditambahkan sehingga mendapatkan total angle yaitu 266. Selanjutnya sistem melakukan asumsi lebar objek dalam satuan milimeter dalam kasus ini memiliki nilai sebesar 750 mm. setelah semua nilai yang dibutuhkan sudah diketahui maka perhitungan jarak dapat dilakukan dengan menggunakan Persamaan (15).

$$\text{Jarak} = \text{Asumsi lebar objek}*(1.0/\text{total angle})*57 / 1000 \text{ [14]} = 750 * (1.0/266)*57 / 1000 = 0.16071 \quad (15)$$

Selanjutnya nilai tersebut diubah ke satuan cm dengan mengalinya dengan 1000 sehingga mendapatkan hasil akhir dari jarak yang terdeteksi yaitu 160 cm.

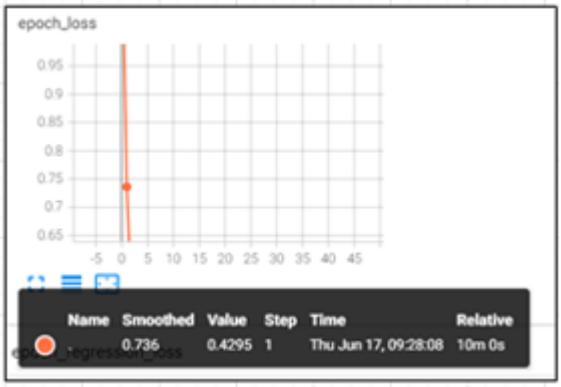


Gambar. 14 Ilustrasi Perhitungan Triangle

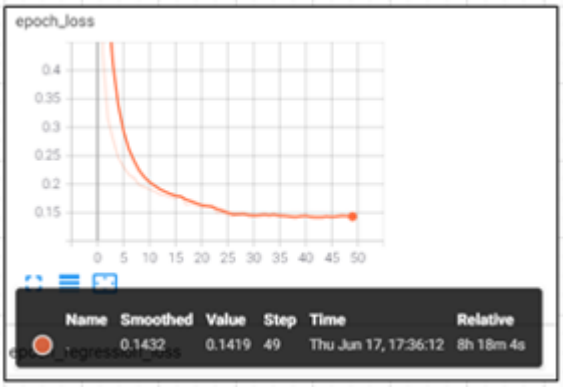
3. HASIL DAN PEMBAHASAN

Pada penelitian ini, proses *training* model *RetinaNet* menggunakan *backbone* arsitektur *ResNet-152*, dengan menggunakan dataset sebanyak 2444 citra. Dataset melalui proses *bounding box* yang berguna untuk memberikan label kelas objek. Proses pemberian label kelas menghasilkan file dengan ekstensi *.xml. Hasil dari *bounding box* kemudian dilakukan proses *training* dengan model *RetinaNet* menggunakan *backbone* arsitektur *ResNet-152* yang menghasilkan file ekstensi *.h5 sebagai model. Proses training yang dilakukan menghasilkan 3 model dengan parameter yang berbeda, model pertama dengan parameter *optimaizer* ADAM, *epoch* 50, *batchsize* 16, dan *learning rate* 1e-5. Model kedua dengan parameter *optimaizer* ADAM, *epoch100*, *batchsize* 16, dan *learning rate* 1e-5. Model ketiga dengan parameter *optimaizer* SGD, *epoch* 50, *batchsize* 16, dan *learning rate* 1e-5.

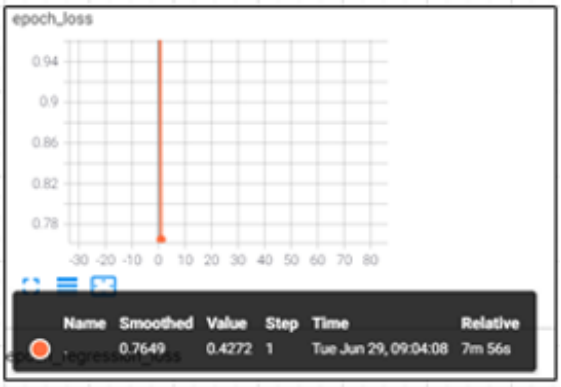
Model pertama yaitu *Optimaizer* ADAM, dengan *epoch* 50, *batchsize* 16, dan *learning rate* 1e-5. Dapat dilihat pada Gambar 15 dan Gambar 16. Pada proses training epoch ke-1 dihasilkan nilai loss sebesar 0.4295 (Gambar 20), pada setiap peningkatan jumlah epoch terjadi penurunan nilai loss. Sehingga nilai loss pada epoch 50 turun menjadi 0.1419.(Gambar 15). Model kedua yaitu Optimaizer ADAM, dengan epoch 100, batchsize 16, dan learning rate 1e-5. Dapat dilihat pada Gambar 17 dan Gambar 18. Pada proses training epoch ke-1 dihasilkan nilai loss sebesar 0.4272 (Gambar 17), pada setiap peningkatan jumlah epoch terjadi penurunan nilai loss. Sehingga nilai loss pada epoch 50 turun menjadi 0.1454.(Gambar 18).



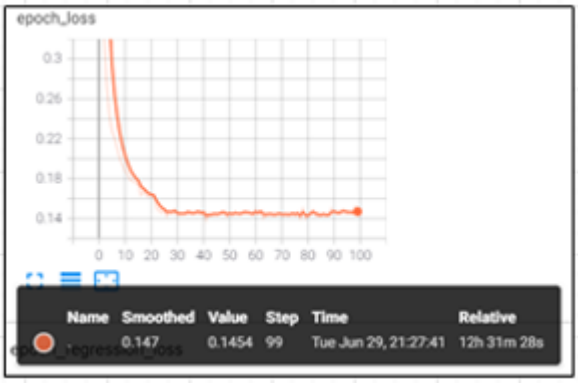
Gambar. 15 Grafik nilai loss epoch ke-1 optimaizer ADAM epoch 50 batchsize 16 learning rate 1e-5



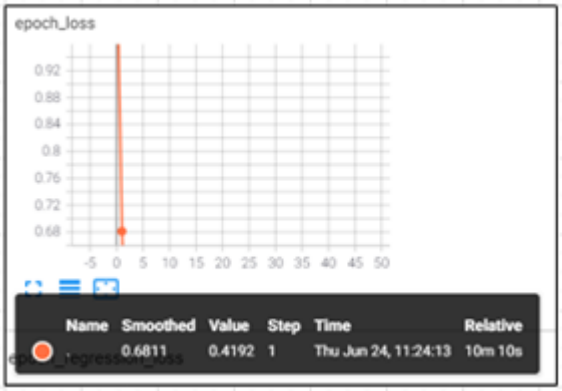
Gambar. 16 Grafik nilai loss epoch ke-50 optimaizer ADAM epoch 50 batchsize 16 learning rate 1e-5



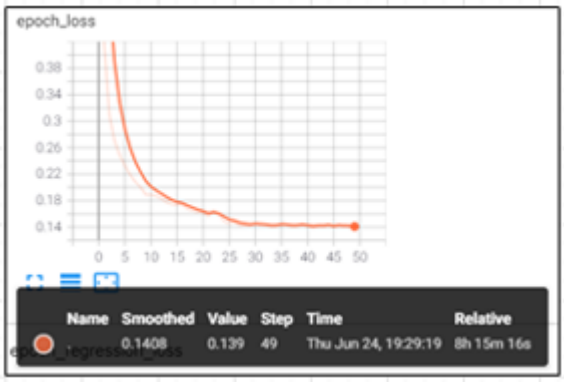
Gambar. 17 Grafik nilai loss epoch ke-1 optimaizer ADAM epoch 100 batchsize 16 learning rate 1e-5



Gambar. 18 Grafik nilai loss epoch ke-50 optimaizer ADAM epoch 100 batchsize 16 learning rate 1e-5

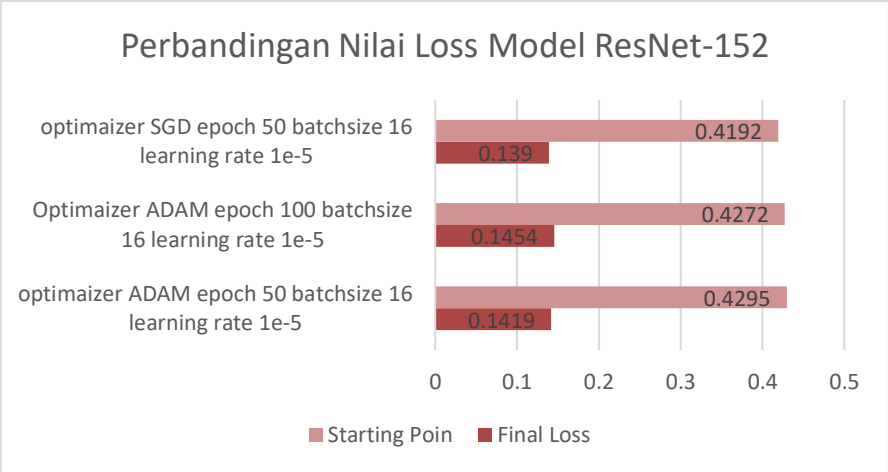


Gambar. 19 Grafik nilai loss epoch ke-1 optimaizer SGD epoch 50 batchsize 16 learning rate 1e-5



Gambar. 20 Grafik nilai loss epoch ke-50 optimaizer SGD epoch 50 batchsize 16 learning rate 1e-5

Model ketiga yaitu *Optimaizer* SGD, dengan *epoch* 50, *batchsize* 16, dan *learning rate* 1e-5. Dapat dilihat pada Gambar 24 dan Gambar 25. Pada proses training epoch ke-1 dihasilkan nilai loss sebesar 0.4192 (Gambar 19), pada setiap peningkatan jumlah epoch terjadi penurunan nilai loss. Sehingga nilai loss pada epoch 50 turun menjadi 0.139.(Gambar 20). Grafik nilai loss pada Gambar 26 menunjukkan pengujian performansi sistem dengan menampilkan nilai loss awal dan nilai loss akhir pada setiap proses training dari 3 parameter yang berbeda. Semakin kecil nilai loss maka model yang dihasilkan dari proses training mendapatkan nilai akurasi yang semakin bagus. Nilai loss dan akurasi memiliki nilai kesatuan yaitu 1 maka model dengan akurasi terbaik yaitu dengan parameter Optimaizer SGD, dengan epoch 50, batchsize 16, dan learning rate 1e-5 yang memiliki nilai loss terkecil yaitu 0.139 dan mendapatkan akurasi sebesar 86.1%



Gambar. 21 Grafik perbandingan nilai loss model ResNet-152

3.1 Pengujian deteksi objek

Pengujian kinerja sistem dilakukan untuk mengukur precision, recall, f1 score dan accuracy dalam mendeteksi objek dengan model RetinaNet menggunakan backbone arsitektur ResNet-152. Data yang digunakan pada saat training dan testing merupakan data citra yang berbeda. Pengujian dilakukan sebanyak 250 citra yang terdiri dari 150 citra objek dan 100 citra yang bukan objek. Pengujian dilakukan terhadap 3 model yang sudah ditraining sebelumnya. Pada table 1 ditunjukkan hasil nilai precision, recall, f1 score, dan accuracy dalam mendeteksi objek dengan model RetinaNet dengan arsitektur ResNet-152. Tabel pengujian deteksi objek menampilkan parameter terbaik dalam mendeteksi objek adalah parameter objek dengan Optimaizer ADAM, Epoch 50, Batchsize 16, Learning Rate 1e-5 dengan akurasi 86%. Hasil ini selaras dengan akurasi yang didapat pada proses training yaitu sebesar 86%

Tabel. 2 Pengujian kinerja sistem deteksi objek menggunakan ResNet-152

No	Pengujian Model	Precision	Recall	F1 Score	Accuracy
1	Optimaizer ADAM, Epoch 50, Batchsize 16, Learning Rate 1e-5	82 %	98%	89%	86%
2	Optimaizer ADAM, Epoch 100, Batchsize 16, Learning Rate 1e-5	77%	96%	85%	81%
3	Optimaizer SGD, Epoch 50, Batchsize 16, Learning Rate 1e-5	80%	98%	88%	84%

3.2 Pengujian Deteksi Jarak

Pengujian deteksi jarak dilakukan untuk mengetahui konfigurasi kamera yang optimal untuk menentukan jarak dengan parameter pengujian ketinggian kamera mulai dari 100-150 cm dan jarak kamera ke objek mulai dari 100-300 cm. Pengujian dilakukan 10 x setiap parameternya, hasil dari pengujian dengan tinggi kamera 150 cm yaitu mendapatkan ketepatan jarak yang memiliki total rata-rata selisih jarak sebesar 57.1 cm. Dari tabel pengujian deteksi jarak ini di dapat bahwa tinggi kamera mempengaruhi ketepatan sistem dalam mendeteksi jarak sehingga diambil kesimpulan bahwa sistem mendapatkan nilai optimal saat tinggi kamera 130 cm karena mendapatkan nilai total selisih paling kecil yaitu 14.3 cm. Berikut ditampilkan hasil dari total selisih setiap pengujian.

Tabel.3 Total Selisih Perhitungan Jarak

Pengujian Tinggi Kamera	Total Selisih
100 Cm	37.0 Cm
110 Cm	54.9 Cm
120 Cm	25.1 Cm
130 Cm	14.3 Cm
140 Cm	31.1 Cm
150 Cm	57.1 Cm

4. KESIMPULAN

Hasil pengujian deteksi objek yang dilakukan dengan menggunakan 250 citra uji yang terdiri dari 150 citra standhanger dan 100 citra yang bukan standhanger kemudian diuji dengan 3 model yang berbeda yaitu model pertama Optimaizer ADAM, dengan epoch 50, batchsize 16, dan learning rate 1e-5. Model kedua dengan Optimaizer ADAM, dengan epoch 100, batchsize 16, dan learning rate 1e-5. Model ketiga yaitu Optimaizer SGD, dengan epoch 50, batchsize 16, dan learning rate 1e-5. Berdasarkan 3 model yang diujikan didapat kesimpulan bahwa model dengan parameter Optimaizer ADAM, Epoch 50, Batchsize 16, Learning Rate 1e-5 menjadi model terbaik dalam mendeteksi objek standhanger dengan nilai precision sebesar 82%, nilai recall sebesar 98%, nilai f1 score sebesar 89% dan nilai accuracy sebesar 86%. Berdasarkan pengujian dalam mendeteksi jarak dengan parameter tinggi kamera diantara 100-150 cm dan jarak objek dari kamera diantara 100-300 cm dengan angle kamera 80º - 90º sehingga diambil kesimpulan bahwa sistem mendapatkan nilai optimal saat tinggi kamera 130 cm karena mendapatkan nilai total selisih paling kecil yaitu 14.3 cm.

REFERENSI

[1] R. R. A. Bourne *et al.*, "Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: a systematic review and meta-analysis," *Lancet Glob. Heal.*, vol. 5, no. 9, pp. e888–e897, 2017, doi: 10.1016/S2214-109X(17)30293-0.

[2] E. Elisawati, "Sistem Deteksi Objek Dengan Menggunakan Sensor Ultrasonik Berbasis Fuzzy," *I N F O R M a T I K a*, vol. 9, no. 1, p. 10, 2018, doi: 10.36723/juri.v9i1.58.

[3] R. O. Wiyagi and M. Y. Mustar, "Deteksi Jarak Objek Bercahaya Secara Real Time Menggunakan Kamera Tunggal," *3rd Indones. Symp. Robot Soccer Compet.*, no. January, pp. 1–5, 2015.

[4] I. S. Pratama, Muhammad Rizky; Rizal, Achmad; Sumaryo, "Desain Sistem Deteksi Objek Real Time Dengan Metode Haar Cascade Classifier Real Time Object Detection System Design Using Haar Cascade Classifier Method," vol. 7, no. 1, pp. 26–34, 2020.

[5] B. Rais, "Alat Bantu Pendeteksi Objek Sekitar Bagi Tuna Netra Menggunakan Stereo Vision Dengan Metode Convolutional Neural Network (CNN)," *Undergrad. thesis*, 2019.

[6] Rahul and B. B. Nair, "Camera-based object detection, identification and distance estimation," *Proc. - 2nd Int. Conf. Micro-Electronics Telecommun. Eng. ICMETE 2018*, pp. 203–205, 2018, doi: 10.1109/ICMETE.2018.00052.

[7] M. I. Arenado, J. M. P. Oria, C. Torre-Ferrero, and L. A. Rentería, "Monovision-based vehicle detection, distance and relative speed measurement in urban traffic," *IET Intell. Transp. Syst.*, vol. 8, no. 8, pp. 655–664, 2014, doi: 10.1049/iet-its.2013.0098.

[8] A. A. Ali, "2016 Al-Sadeq International Conference on Multidisciplinary in IT and Communication Science and Applications (AIC-MITCSA) ±IRAQ (9-10) May," no. 1, 2016.

[9] D. Y. Huang, C. H. Chen, T. Y. Chen, W. C. Hu, and K. W. Feng, "Vehicle detection and inter-vehicle distance estimation using single-lens video camera on urban/suburb roads," *J. Vis. Commun. Image Represent.*, vol. 46, pp. 250–259, 2017, doi: 10.1016/j.jvcir.2017.04.006.

[10] S. R. Dewi, "Deep Learning Object Detection Pada Video," *Deep Learn. Object Detect. Pada Video Menggunakan Tensorflow Dan Convolutional Neural Netw.*, pp. 1–60, 2018, [Online]. Available: https://dspace.uii.ac.id/bitstream/handle/123456789/7762/14611242_Syarifah Rosita Dewi_Statistika.pdf?sequence=1.

[11] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[13] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, 2020, doi: 10.1109/TPAMI.2018.2858826.

[14] B. Dipper, "Measuring Size from Images: A wrangle with angles and image scale," *Observatory*