J-Icon : Jurnal Komputer dan Informatika
Vol. 11 No. 2, Oktober 2023, pp. 145~156
DOI: 10.35508/jicon.v11i2.9903

Residual Network Layer Comparison For Seat Belt Detection

Irma Amelia Dewi[1], Nur Zam Zam Nasrulloh[2]

[1,2]Institut Teknologi Nasional Bandung, Jl. Phh. Mustafa No. 23 Bandung, Indonesia
[1]Email: irma_amelia@itenas.ac.id

ABSTRAK

Sebagian besar pemantauan pelanggaran di jalan raya Indonesia saat ini dilakukan secara manual dengan memantau melalui kamera CCTV sehingga pengendara mobil masih melakukan kemungkinan pelanggaran dalam penggunaan sabuk pengaman. Residual Network (ResNet) sebagai salah satu arsitektur dengan tingkat akurasi mencapai 96.4% pada tahun 2015, yaang ditujukan untuk mengatasi masalah vanishing gradient yang biasa terjadi pada jaringan dengan layer yang banyak. Oleh karena itu, dalam penelitian ini, sistem dikembangakan menggunakan arsitektur RetinaNet untuk untuk mendeteksi pengemudi yang menggunakan sabuk pengaman dan pengemudi yang tidak menggunakan sabuk pengaman dengan bacbone ResNet. Selain itu pada penelitian ini membandingkan kinerja dari performansi ResNet-101 dan ResNet-152. Adapun hyperparameter yang digunakan diantaranya jumlah dataset sebanyak 10,623 citra pada proses pelatihan, dan parameter batch size adalah 1, dengan jumlah 10,623 steps, dan jumlah epoch sebanyak 16. Berdasarkan 60 pengujian yang dilakukan pada penelitian ini, model RetinaNet dengan arsitektur ResNet-152 melakukan lebih baik daripada arsitektur ResNet-101. Arsitektur ResNet-152 menghasilkan performa sistem dengan akurasi 98%, nilai presisi 99%, nilai recall 99%, dan skor F1 99%.
Kata kunci: sabuk pengaman, ResNet, CNN, deteksi objek

ABSTRACT

Most of the monitoring of traffic violations on Indonesian roads is currently done manually by monitoring through CCTV cameras, so drivers still have the possibility of violating the use of seat belts. Residual Network (ResNet) as one of the architectures with an accuracy rate of up to 96.4% in 2015, which is intended to overcome the vanishing gradient problem that commonly occurs in networks with many layers. Therefore, in this study, a system was developed using the RetinaNet architecture to detect drivers who use seat belts and drivers who do not use seat belts with the ResNet backbone. In addition, this study compares the performance of ResNet-101 and ResNet-152. The hyperparameters used include a dataset of 10,623 images in the training process, and the batch size parameter is 1, with a total of 10,623 steps, and the number of epochs is 16. Based on 60 tests conducted in this study, the RetinaNet model with the ResNet-152 architecture performed better than the ResNet-101 architecture. The ResNet-152 architecture resulted in a system performance with an accuracy of 98%, precision value of 99%, recall value of 99%, and an f1 score of 99%.
Keywords: seat belt, ResNet, CNN, object detection

1. INTRODUCTION

The seat belt is one of the tools that both drivers and passengers must use to maintain driving safety. The warning system of safety belt usage is also an obligatory feature, which reminds the driver to use a seat belt [1]. Every driver and car passenger on the road is required to use a seat belt. The seat belt can reduce the risk of fatal injury to the driver by 45% and the risk of moderate to critical damage by 50% [2]**.** However, many drivers underestimate the importance of seat belt use in the safety of driving on the highway, causing accidents. Even the government has tightened the rules by imposing sanctions on road users' violations, especially those who do not use seat belts. Based on Law No. 22/2009 on road traffic and transportation, article 289, if someone does not wear a safety belt, the punishment is imprisonment of one month (maximum penalty) or a maximum fine of Rp. 250,000.

Currently, safety-supporting technology related to the use of seat belts is growing. All aim to increase the safety factor for vehicle users in emergency conditions, with the development of artificial intelligence technology and the widespread implementation of deep learning methods. So in this research, the detection of seat belts for car drivers on the road has been carried out using the deep learning method.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Deep learning is a method that is composed of multilevel layers for detection, segmentation and classification of objects with multilevel abstraction levels [3]. There is an algorithmic approach in the deep learning method, such as the Convolutional Neural Network (CNN) for object classification/detection [4] . CNN is one of the most prominent deep learning methods, where multiple layers are trained powerfully. This method is effectively and commonly used in computer vision applications [5]. With the development of computer vision applications, developed methods CNN model RetinaNet architecture with a goal in classification/detection becomes more accurate in its performance.

RetinaNet has accurate performance exceeding two-stage detectors in focal loss and training data [6]. RetinaNet can work in various types of backbone network architecture from CNN, such as Residual Network (ResNet) (ResNet-50, ResNet-101, ResNet-152), VGG net-16, VGG net-19 dan DenseNet [7]. In the annual competition held by the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) and Common Objects in Context (COCO), various types of architectural backbones in classifying / detecting objects were introduced. In 2015, the first winner of the ILSVRC and COCO competitions by RetinaNet Model used the ResNet-152 architecture backbone, which had the lowest error rate of 3.6%, as shown in Figure 1 [8] [9].

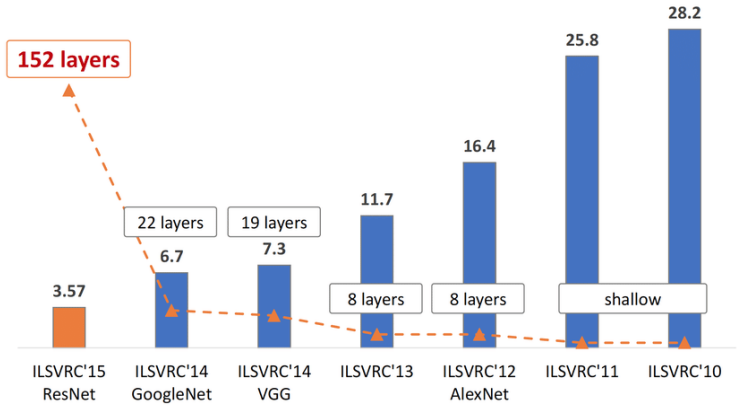

Figure 1. Annual competition ILSVRC [8]

Therefore, this research aims to compare system performance accuracy with the RetinaNet model using ResNet-101 and ResNet-152 architecture backbone to detect car drivers' seat belts. The gap of the research is the development of a system using RetinaNet architecture with the ResNet bacbone to detect drivers who use seat belts and drivers who do not use seat belts. The study compares the performance of ResNet-101 and ResNet-152 in detecting seat belt usage, using a dataset of 10,623 images in the training process, and the batch size parameter is 1, with a total of 10,623 steps, and the number of epochs is 16. This research aims to improve the accuracy of seat belt detection in real-time traffic monitoring systems by utilizing deep learning methods, which can contribute to improving road safety.

2. RESEARCH METHOD

There are 2 parts to the block diagram process, namely the training and testing process, can be seen in Figure 2. In the training data process, images of car drivers who use seat belts and without seat belts are collected. These images are labeled using LabelImg to become coordinate object data (xmin, xmax, ymin, ymax) and object class labels that are stored in files with *.xml extension [10]. The output produced in the training process is the RetinaNet Model using the ResNet-101 and ResNet-152 architecture separately, saved with the file *.h5 extension.

In the testing process, system input is in the form of a front view car video recording file. The input video is extracted into several image frames. For each frame, a preprocessing process is carried out to create zero paddings for each color channel by reducing the BGR image matrix with a Caffe mode filter. Then, the process of feature map extraction is carried out using the ResNet-101 and ResNet-152 architecture. The architecture of RetinaNet can be seen in Figure 3. RetinaNet is a network consisting of one backbone to calculate the feature map convolutionally on all images and two subnetworks. The first subnetwork functions to classify objects, and the second subnetwork functions to form a bounding box regression. Each level of the pyramid can be used to detect objects on a different scale. Feature Pyramid Network (FPN) [11] improves predictions at multiple scales on fully connected networks (FCN) [12]. Figure 3 shows the RetinaNet architecture layer.

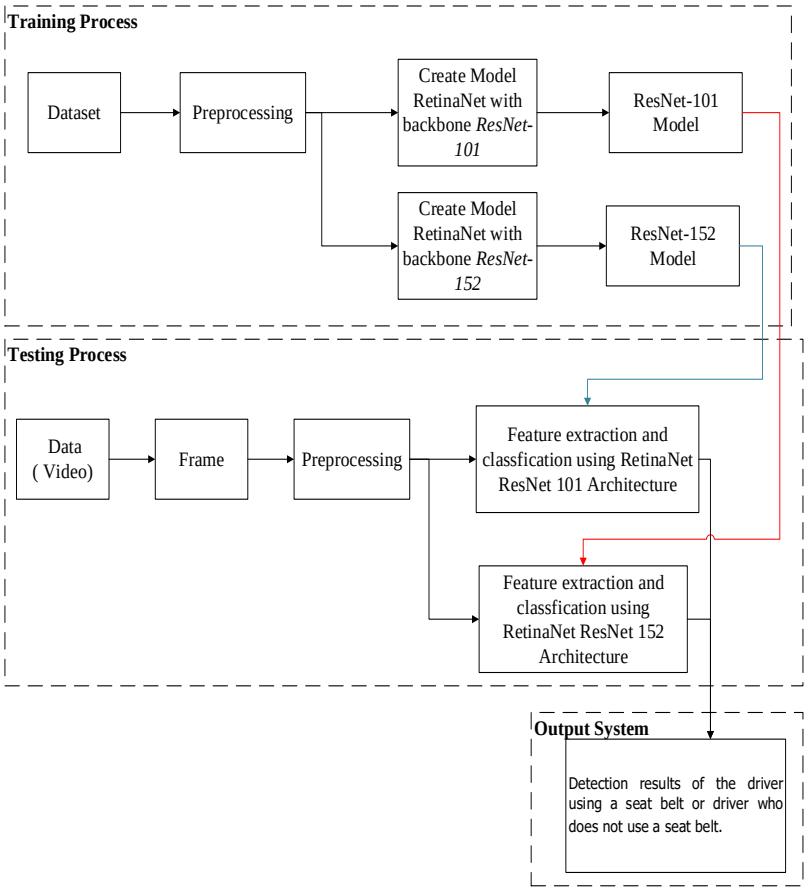

Figure 2. Block diagram system

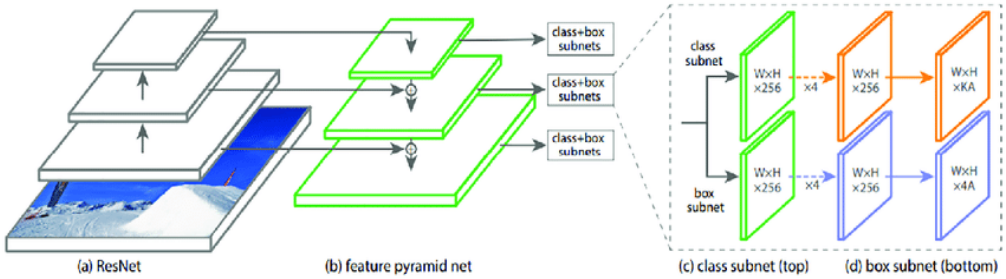

Figure 3. RetinaNet architecture [12]

Residual Network (ResNet) is a residual network that has deep networks. The deepest network of ResNet is 152 layers. ResNet architecture has five layers, the first layer is 18, the second layer is 34, the third layer is 50, the fourth layer is 101, and the fifth layer is 152. Each layer has a different number of convolution depths and produces a feature map/weight for object detection based on the dataset owned [13]. Figure 4 shows the ResNet architecture layer.

In the ResNet architecture, a convolution process is carried out. The convolution process results are the feature map/weight value of the test data by producing anchor predictions to detect objects. Figure 5 shows an illustration of the anchor box prediction for detecting objects. An anchor box is a set of bounding boxes that have been determined with a certain height and width. Anchor boxes are defined to capture the scale and aspect ratio of detected object classes and are usually selected based on the objects' size in the training data set[14][15][16]. The anchor area ranges from 322 to 5122 at each level of the pyramid with an aspect ratio of {1:2, 1:1, 2:1}.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				

Figure 4. ResNet Architecture and Layers [13]

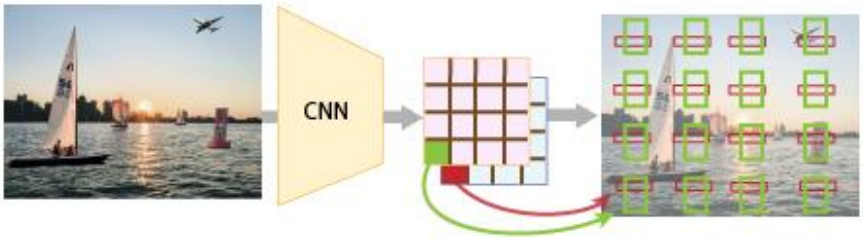

Figure 5. Illustration of Anchor Box [15]

The following process is the box regression process and the classification process. The box regression process is carried out to regress any excess value on the detected object's bounding box. Then the classification process is carried out to classify the object of the driver who uses a seat belt and does not use a seat belt to produce a value and object class label.

Preprocessing

The preprocessing calculation of every color channel's original image matrix RGB (Red, Green and Blue) with the Caffe Mode kernel using Equation 1 to produce a preprocessing matrix. A similar process is carried out on the Red (R) and Green (G) color channels.

$$\begin{matrix}[B] \\ [G] \\ [R]\end{matrix} = \begin{matrix}[B] \\ [G] \\ [R]\end{matrix} - \begin{matrix}[103{,}939] \\ [116{,}779] \\ [123{,}68]\end{matrix} \quad \text{..... (1)}$$

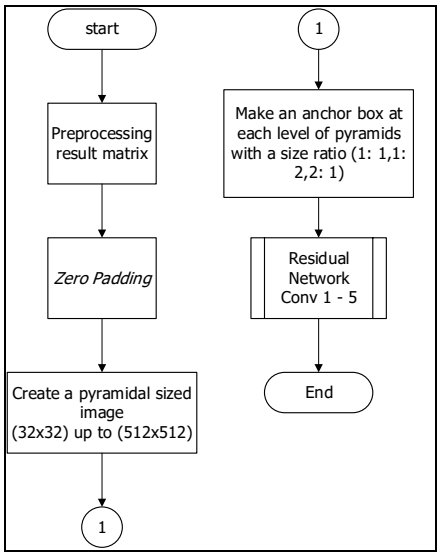

Figure 6. Feature Map Extraction

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Feature Map Extraction Process

The feature extraction process using Model RetinaNet with ResNet 101 and ResNet 152 backbone can be seen in Figure 6. A pyramid with sizes 322, 642, 1282, 2562, 5122 is made at this stage, as shown in Figure 3. The higher the pyramidal level, the smaller the image resolution.

1. Zero-Padding

Each anchor aims to predict the existence of an object. A zero-padding process is carried out at each anchors adding a matrix dimension to the image's side with the number 0, so the image matrix dimensions are bigger [17].

2. Pyramid Network Feature Process and anchor box

After the zero-padding process is carried out, the first step is to make a pyramid with sizes 322, 642, 1282, 2562, 5122. The higher the pyramidal level, the less image resolution. Each level of the pyramid has a different size and scale. Then at each level of the pyramid, an anchor is made with a ratio of {1:2, 1:1, 2:1}. Figure 7 illustrates an anchor box's with a set of boxes in red on the driver object's image using a seat belt.

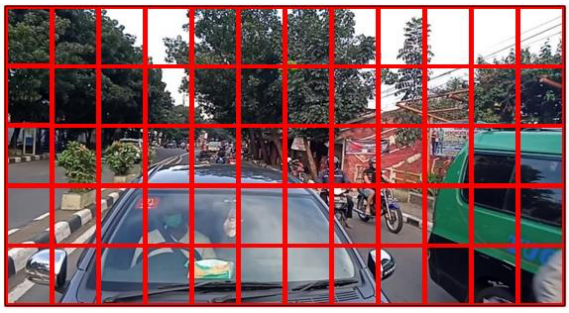

Figure 7. Illustration of the Anchor Box in the image

Each anchor is processed using convolution with ResNet-101 and ResNet-152 architectural backbone to produce feature map/weight values to predict the existence of recognized objects based on the dataset.

3. Residual Network Process

The following is the residual network subprocess. The ResNet process uses 101 layers and 152 layers. The process consists of 7x7 convolution operations, 3x3 max pooling, 1x1 convolution, ReLU activation, 3x3 convolution operations, ReLU activation, 1x1 convolution operation [13][18]. The number of filters used in each convolutional operation on the residual module is adjusted to the residual network layer level. The ResNet process generates a feature map/weight value to predict the driver objects that use seat belts and those that didn't use seat belts based on the dataset model that has been created.

Residual network process described in Figure 8, Figure 9, Figure 10, Figure 11, and Figure 12. The residual processes of ResNet 101 and ResNet 152 have a different number of filters. The repetition of the multiplication process varies according to the number of layers used in the ResNet architecture. As in Figure 9, the value of i = 3 illustrates the process of repeating a 3x convolution.

4. Convolution and Max Pooling Process

In Figure 8, the initial residual network process is a convolutional process using a 7x7 kernel matrix with a shift of 2 strides. The convolution process multiplies the input image with the kernel or filter to get features on the image [19][20]. The Equation of the convolution process can be seen in Equation 2,

$$h(x,y) = f(x,y) * g(x,y) = \sum_{a=-\infty}^{\infty} \sum_{b=-\infty}^{\infty} f(a,b) * g(x-a, y-b) \quad \text{.....(2)}$$

where h(x,y) is a pixel of convolution result, f(x,y) is a pixel of the original image, and g(x,y) is a kernel or filter template.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

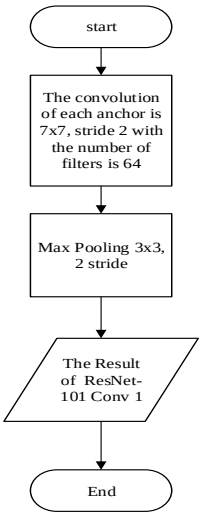

Figure 8. The Resnet Process Flow (1)

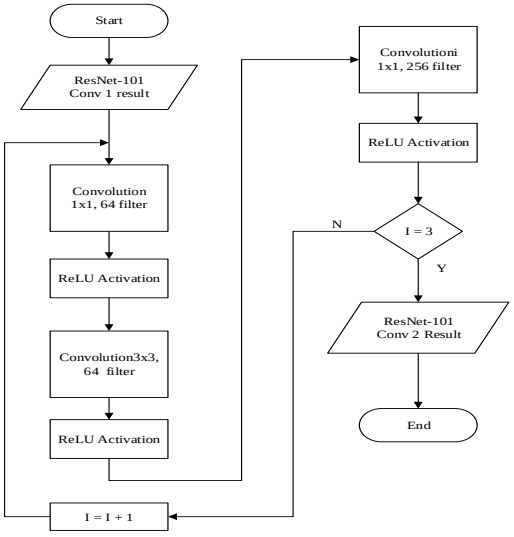

Figure 9. The Resnet Process Flow (2)

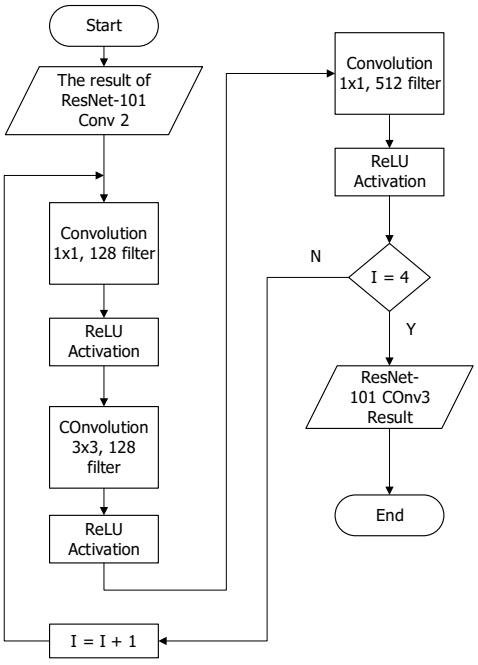

Figure 10. The Resnet Process Flow (3)

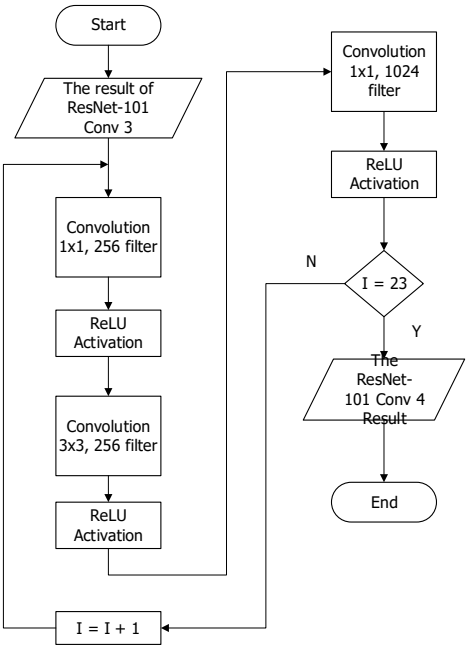

Figure 11. The Resnet Process Flow (4)

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

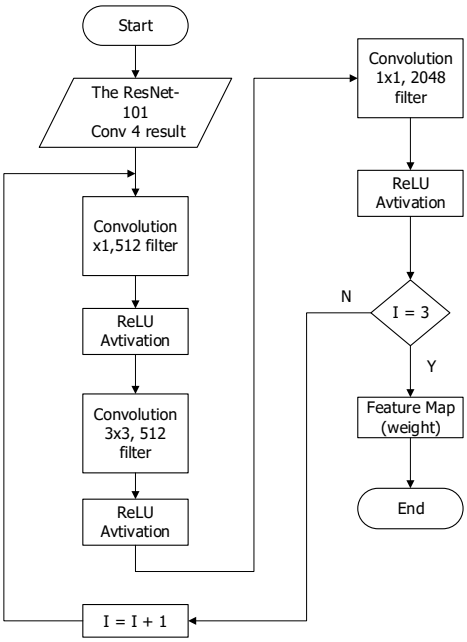

Figure 12. The Resnet Process Flow (5)

The max-pooling process was carried out with a 3x3 matrix and 2 stride shifts from the results of the convolution process. As an example of a 4x4 convolution matrix, the max-pooling process is carried out with a 2x2 kernel matrix size and 2 stride shifts, then taking the maximum value for each image pixel.

5. Regression Process

The regression process is carried out to regress any excess value on the detected object's bounding box as shown in Figure 13. The first step is to convolute 3x3 the feature map results three times with 256 filters. The ReLU activation process to change the resulting negative (-) value to 0. The ReLU activation process is calculated using Equation 3.

$$ReLu(x) = max(0, x) \quad \text{..... (3)}$$

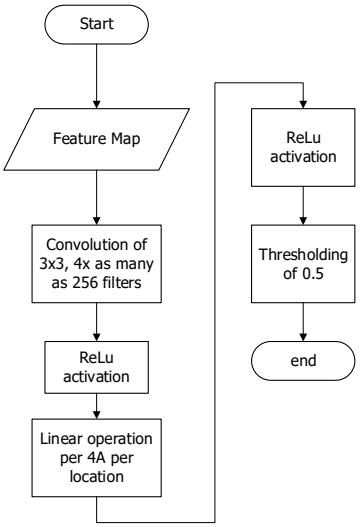

Figure 13. Regression Box

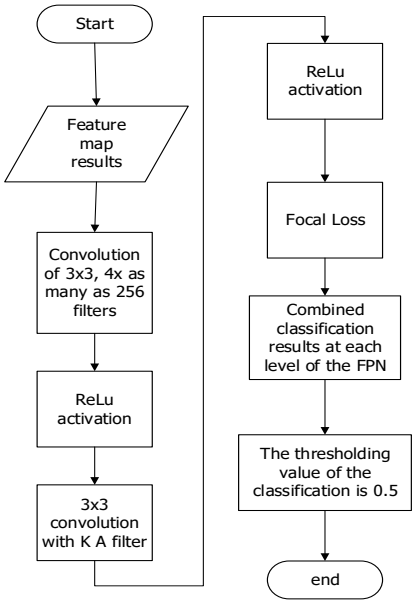

Figure 14. Classification Process

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

The next step is a linear operation for every four operations (A = anchor) per spatial location. Every four anchors A per spatial location can predict the relative offset between the anchor and the ground-truth box. In the fifth stage, the ReLU activation process is carried out using Equation 3. In the sixth stage, a non-maximum suppression function is performed. Each anchor predicted by the object is given a thresholding score of 0.5. If the confidence score <0.5, the bounding box is deleted. If the confidence score is > 0.5, then a bounding box is generated, predicted by the object with the highest score. The bounding box that has the highest score is the object successfully classified. The confidence score is obtained from the results of each anchor, which is predicted as an object.

6. Classification Process

Figure 14 shows the flowchart of the classification process. The classification process is carried out to detect the driver's object using a seat belt and a driver who did not use a seat belt to produce a value and object class label.

The classification process's initial stage is the convolution of the feature extracted image with a 3x3 kernel four times with 256 filters. Then, ReLU activation is carried out to change the negative (-) value to 0. In the next step is convolution 3x3 with filter K = class and A = anchor. Next, the ReLU activation process is carried out again and continues with the focal loss process to calculate the loss value on the detected class's ground-truth label. In the focal loss process, a parameter with a value of γ used is 2, and the value of α used is 0.25 to get maximum results on the use of the focal function [6]. Focal loss can process with Equation 4.

$$FL(pt) = -\alpha(1 - pt)\gamma \, log(pt) \quad (4)$$

3. RESULT AND DISCUSSION

Training Process

In this study, the training process used a dataset of objects using seat belts totaling 10,623 images. The dataset used consists of a bounding box label for the object class of the driver who uses a seat belt (SP) and the driver who does not use a seat belt (NSP), as in Figure 15. Every image consists of coordinates and object's class, which is then stored in a file with the extension *. xml. The training process uses parameters with a batch size of 1, 10,623 steps, and 16 epochs. The training process's output results produce loss values, accuracy values, and * .h5 extension.

This training process utilizes Google Colaboratory with the Nvidia Tesla V100 SXM2 GPU's hardware specifications with 25 GB RAM Memory. Figure 16 and Figure 17 show a graph of the loss value from the RetinaNet training model results with the ResNet-101 and ResNet-152, respectively.

Figure 15. Example of a driver's object dataset using seat belts

Figure 16. ResNet-101 Training Loss Graph

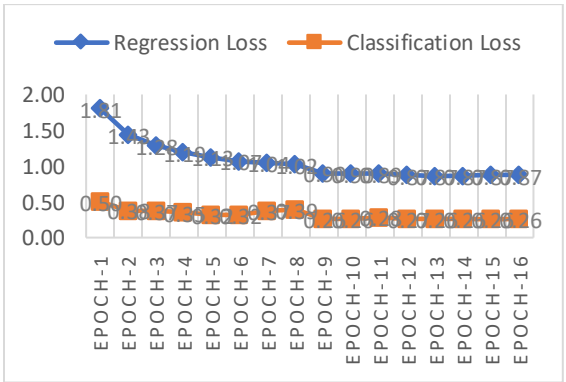

Figure 17. ResNet-152 Training Loss Graph

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Based on Figure 16, the loss value from the RetinaNet training model results with the ResNet-101 architecture. In the 1st epoch training process, the regression loss was 1.8057, and the classification loss was 0.4847. At each increase in the number of epochs, there was a decrease in the loss value. In the 16th epoch, the loss value decreased with regression loss of 0.8576, classification loss of 0.2669. Meanwhile based on Figure 17, the 1st epoch of the ResNet-152 training process resulted in a regression loss of 1.8053, a classification loss of 0.5026. At each increase in the number of epochs, there was a decrease in the loss value. At the 16th epoch, the loss value decreased with regression loss of 0.8678, classification loss of 0.2623.

The training process also measures accuracy, from the RetinaNet training model's results with the ResNet-101 architecture obtained in the 1st epoch training process, the regression accuracy is 0.4301, and classification accuracy is 0.7018. With each increase in the number of epochs, the classification accuracy value increases. In the 16th epoch, there was an increase in classification accuracy with regression accuracy of 0.3941 and classification accuracy of 0.9901. Meanwhile, the results of the RetinaNet training model with the ResNet-152 architecture. In the 1st epoch training process, the regression accuracy is 0.4283, and the classification accuracy is 0.8010. With each increase in the number of epochs, the classification accuracy value increases. In the 16th epoch, there was an increase in classification accuracy with regression accuracy of 0.3803 and classification accuracy of 0.9939. Based on the results of training using ResNet-101 and ResNet-152, the following comparisons were obtained:

Table 1. Comparison Table of Training Results using ResNet-101 and ResNet-152

Backbone	Regression Loss	Classification Loss	Regression Accuracy	Classification Accuracy
ResNet-101	0.86	0.27	0.39	0.99
ResNet-152	0.87	0.26	0.38	0.99

The ResNet-101 and ResNet-152 backbones have the same performance from the training process comparison table without any significant differences, as shown in Table 1.

Testing Process

The detection test for seat belts was carried out at traffic light stops on Jalan Suci and Jalan Soekarno Hatta, Bandung. Tests carried out are as many as 60 images on the object of the driver who uses a seat belt (SP) and the driver who does not use a seat belt (NSP). In the testing process, system input is in the form of a front view car video recording file. The video is then extracted into frames. A preprocessing process is carried out in each frame to create zero paddings for each color channel; Figure 18 shows the preprocessing process' image. Then the feature map extraction process is carried out with the RetinaNet model using the ResNet architecture so that the feature map/weight values are obtained from the test data by producing an anchor prediction to detect the driver using a seat belt, in Figure 19 is the image of the feature map extraction process using ResNet.

Furthermore, the box regression and classification processes are carried out. This process is carried out simultaneously to produce labels, bounding boxes, and classification values on detected objects. The box regression process is carried out to regress any excess value on the bounding box of the detected object, as shown in Figure 20 the image of the box regression process is shown. Then, the classification process is carried out to detect the existence of a driver object using a seat belt and a driver who does not use a seat belt, as shown in Figure 21 an image of the classification results of a driver using a seat belt is shown.

Figure 18. Preprocessing Results

Figure 19. The Feature Map Extraction Result

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Figure 20.Regression Box Result

Figure 21. Classification Result

System performance testing is carried out by measuring precision, recall, f1 score, and accuracy [21][22]. Precision is the level of accuracy between the ground truth and the results given by the system. The recall is the success rate of the system in finding information. F1 Score is the comparison of the average precision value with the weighted recall value. Accuracy is the level of closeness between the predicted value and the real value.

The test was carried out 60 times which were divided into two stages. The first stage of testing was carried out on 32 images of driver objects using seat belts. The second stage of testing was carried out on 28 images of the driver who did not use a seat belt. Table 2 shows the results of the average precision, recall, F1 score, and accuracy in detecting the use of seat belts in car drivers using the ResNet-101 and ResNet-152 architecture.

Table 2. Testing system performance using ResNet-101 and ResNet-152

No	Testing	Resnet-101				ResNet-152			
		Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score	Accuracy
1	Image object of a driver wearing a seatbelt	97%	97%	97%	97%	100%	100%	100%	100%
2	Image object of a driver not wearing a seatbelt	77%	77%	76%	75%	98%	98%	98%	96%
	Average	87%	87%	87%	86%	99%	99%	99%	98%

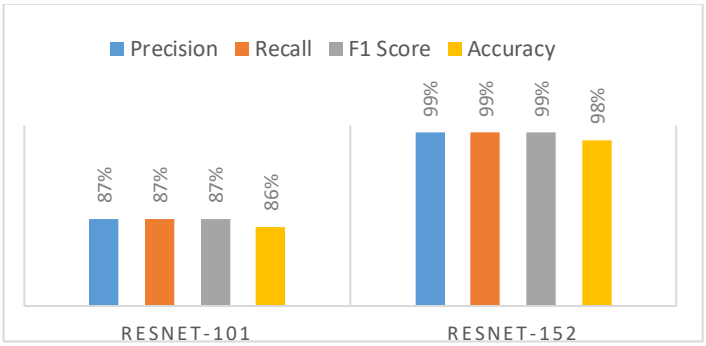

Figure 22. Performance System Comparison Chart of ResNet-101 and ResNet-152

Figure 22 shows the seat belt use detection system's test results for car drivers with the RetinaNet model using ResNet-101 and ResNet-152 architectures. In testing, this system shows that the ResNet-152 architecture is better than the ResNet-101 architecture in detecting the use of seat belts in car drivers. The RetinaNet model uses the ResNet-152 architecture in detecting the use of seat belts showing a precision value of 99%, a recall value of 99%, an f1 score of 99%, and an accuracy value of 98%.

In testing this system, the ResNet-152 architecture is better at detecting seat belt usage because the ResNet-152 architecture has a deeper feature extraction process for object detection processes. The deeper the feature extraction process on ResNet, the more accurate the system will be in detecting driver objects using seat belts and driver objects that do not use seat belts.

4. CONCLUSION

Based on the results of the research that has been done, there are several conclusions. In the RetinaNet model's training process using the ResNet-101 backbone architecture, the regression loss value is 0.8576, the classification loss is 0.2669, the regression accuracy is 0.3941, and the classification accuracy is 0.9901. With the RetinaNet model using the ResNet-152 architectural backbone, the regression loss value

is 0.8678. Classification loss of 0.2623, regression accuracy of 0.3803, and classification accuracy of 0.9939. Decreasing the loss value can increase the classification results on the object. Based on the results of these comparisons in the training process, ResNet-101 and ResNet-152 do not have a significant difference. Based on the test, in detecting seat belts in car drivers with the RetinaNet model, using the ResNet-152 architecture is better than the ResNet-101 architecture. The ResNet-152 architecture shows a precision value of 99%, a recall value of 99%, and f1 score of 99%, and an accuracy value of 98%. There is limitation of this research that should be acknowledge. The dataset used in this study only consist of images taken from a single location and might not represent the diversity of driving conditions in Indonesia. To address the limitation, future research colud consider expanding the dataset to include images from various locations and driving conditions, as well as adding other safety-related issues in driving.

REFERENCES

[1] A. Kashevnik, A. Ali, I. Lashkov and N. Shilov, "Seat Belt Fastness Detection Based on Image Analysis from Vehicle In-Cabin Camera," in *2020 26th Conference of Open Innovations Association (FRUCT)*, Yaroslavl, Russia, 2020, doi: 10.23919/FRUCT48808.2020.9087474

[2] E. Snyder, "Seat Belt Statistic," A Personal Injury Law Firm Representing Injured People, 2019. [Online]. Available: https://www.edgarsnyder.com/car-accident/defective-products/seat-belts/seat-belts-statistics.html. [Accessed 21 October 2020].

[3] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Computer Science*, vol. 2, no. 420, pp. 1-20, 2021, doi: 10.1007/s42979-021-00815-1.

[4] Z.Q. Zhao, Z.Q. Zhao, P. Zheng, S.-T. Xu and X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212 - 3232, 2019, doi: 10.1109/TNNLS.2018.2876865.

[5] W. Chen, Y. Liu, W. Wang, E. M. Bakker, T. Georgiou, P. Fieguth, L. Liu and M. S. Lew, "Deep Learning for Instance Retrieval: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1-20, 2022, doi: 10.1109/TPAMI.2022.3218591.

[6] T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Doll´ar, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. Volume: 42, no. Issue: 2, pp. 318 - 327, 2020.

[7] T. M. Hoang, P. H. Nguyen, N. Q. Truong, Y. W. Lee and K. R. Park, "Deep RetinaNet-Based Detection and Classification of Road Markings by Visible Light Camera Sensors," *Sensors*, vol. 19, no. 2, pp. 1-25, 2019, doi: 10.3390/s19020281.

[8] Á. Arcos-García, J. A. Alvarez-Garcia and L. M. S. Morillo, "Evaluation of Deep Neural Networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332-344, 2018, doi: 10.1016/j.neucom.2018.08.009.

[9] X. Ding, Z. Lin, F. He, Y. Wang and Y. Huang, "A Deeply Recursive Convolutional Network For Crowd Counting," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, 2018, doi: 10.1109/ICASSP.2018.8461772.

[10] N. A. Rahmad, N. A. J. Sufri, N. H. Muzamil and M. A. As'ari, "Badminton player detection using faster region convolutional neural network," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 14, no. 3, p. 1330 – 1335, 2019, doi: 10.11591/ijeecs.v14.i3.pp1330-1335.

[11] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, 2017.

[12] H. Jung, B. Kim, I. Lee, M. Yoo, J. Lee, S. Ham, O. Woo and J. Kang, "Detection of masses in mammograms using a one-stage object detector based on a deep convolutional neural network," *Journal Plos One*, pp. 1-16, 2018, doi: 10.1371/journal.pone.0203355.

[13] K. He , X. Zhang, . S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv*, pp. 1-12, 2015.

[14] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li and J. Shi, "FoveaBox: Beyound Anchor-Based Object Detection," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 29, pp. 7389-7398, 2020, doi: 10.1109/TIP.2020.3002345.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

[15] MathWorks, "Anchor Boxes for Object Detection," 2020. [Online]. Available: https://www.mathworks.com/help/vision/ug/anchor-boxes-for-object-detection.html.

[16] I. A. Dewi, L. Kristiana, A. R. Darlis and R. F. Dwiputra, "Deep Learning RetinaNet based Car Detection for Smart Transportation Network," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi, & Teknik Elektronika*, vol. 7, no. 3, p. 570 – 584, 2019, doi: 10.26760/elkomika.v7i3.570.

[17] M. Hashemi, "Enlarging smaller images before inputting into convolutional neural network: zero-padding vs. interpolation," *Journal of Big Data*, pp. 1-13, 2019, doi: 10.1186/s40537-019-0263-7.

[18] J. Liang, "Image classification based on RESNET," in *The 2020 3rd International Conference on Computer Information Science and Application Technology (CISAT)*, Dali, China, 2020, doi: 10.1088/1742-6596/1634/1/012110.

[19] M. A. Hossain and M. S. A. Sajib, "Classification of Image using Convolutional Neural Network (CNN)," *Global Journal of Computer Science and Technology Neural & Artificial Intelligence*, vol. 19, no. 2, pp. 12-18, 2019, doi: 10.34257/GJCSTDVOL19IS2PG13.

[20] R. Munir, "Konvolusi dan Transformasi Fourier," in *Pengolahan Citra Digital*, Bandung, Informatika, 2004 , pp. 61-73.

[21] T. Karlita, I. M. G. Sunarya, J. Priambodo, R. Rokhana, E. M. Yuniarno, I. K. E. Purnama and M. H. Purnomo, "Deteksi Region of Interest Tulang pada Citra B-mode secara Otomatis Menggunakan Region Proposal Networks," *JNTETI (Jurnal Nasional Teknik Elektro dan Teknologi Informasi)*, pp. 68-76, 2019. [Online]. Available: https://journal.ugm.ac.id/v3/JNTETI/article/view/2618. [Accessed 21 October 2020].

[22] F. H. K. Zaman, J. Johari and A. I. M. Yassin, "Learning Face Similarities for Face Verification using Hybrid Convolutional Neural Networks," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, no. 3, pp. 1333 - 1342, 2019, doi: 10.11591/ijeecs.v16.i3.pp1333-1342.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)