Real Time Detection Of Waste Type Using Single Shot Multibox Detector

Abdiansah Abdiansah [a,1], M. Qurhanul Rizqie [a,2], M. Hatta Aldino Ramadhan [a,3]

[a] Artificial Intelligence Laboratory, Universitas Sriwijaya, South-Sumatera, Indonesia
[1] abdiansah@unsri.ac.id; [2] qurhanul.rizqie@ilkom.unsri.ac.id; [3] mhattaldino@gmail.com

ARTICLE INFO

Article history
Received 23 Nov 2021
Revised 5 Dec 2021
Accepted 10 Dec 2021

Keywords
Detection
Waste Type
Real Time
Single Shot Multibox Detector
SSD300
SSD512

ABSTRACT

The lack of human initiative to manage their own wastes is one of many reasons why waste management in residential area is not optimal. A system to detect waste type in real time is a necessity to support the waste management process to be faster and optimal. This research propose the waste type detection systems using 2 types of Single Shot Multibox Detector models, SSD300 and SSD512. Both models were compared based on the accuracy and speed of detection on TACO dataset dan Waste Classification Data. SSD512 achieves a better accuracy of 0.63 mAP compared to the accuracy of SSD300, which is 0.57 mAP. Both models can also be said to be real time, with the SSD300's detection speed being faster at 51 fps compared to the SSD512's detection speed at 28 fps.

1. Introduction

The lack of people awareness and application of old methods have found it difficult to manage large volumes of solid waste caused by urban population growth [1]. The consequences of this uncontrolled waste can occur in the environment, both the impact on the physical and chemical components (water and air quality) and biology, socio-economics, culture, and health. Hence it is necessary to have a system that can help to to manage waste automatically and quickly.

To make a system that can detect waste type in real time, Single Shot Multibox Detector (SSD) is one of tool that suitable for this purpose. SSD is the first object detector that doesn't use additional processes or networks to initialize bounding boxes, instead it's generated manually. It makes the detection of bounding boxes and score for each labels happen directly in one process. And to maintain the accuracy, SSD performs detection at various scales and aspect ratios [2].

One that affects the performance of the SSD is the base network used to extract feature map. In this research, the initial SSD concept using VGG-16 as base network will be maintained to create a system that can detect waste type accurately in real time. Since VGG-16 has a large number of parameters, the network should be modified to make the entire detection process faster.

There are 2 models of SSD tested in this research, SSD300 and SSD512. Both models are compared based on its accuracy and speed of detection to determine which model has better accuracy while being able to detect in real time.

2. Literature Study / Hypotheses Development

2.1. Dataset

The data used is secondary data in the form of 2-dimensional images containing various solid waste objects which are divided into 2 classes, organic waste and inorganic waste. The data is a combination of data taken from the TACO dataset [3] and Waste Classification Data from Kaggle. Some of this data is obtained from image data extraction on ImageNet [4] and verified community image collection . The total data used is 5,000 images (4000 training data and 1000 testing data).

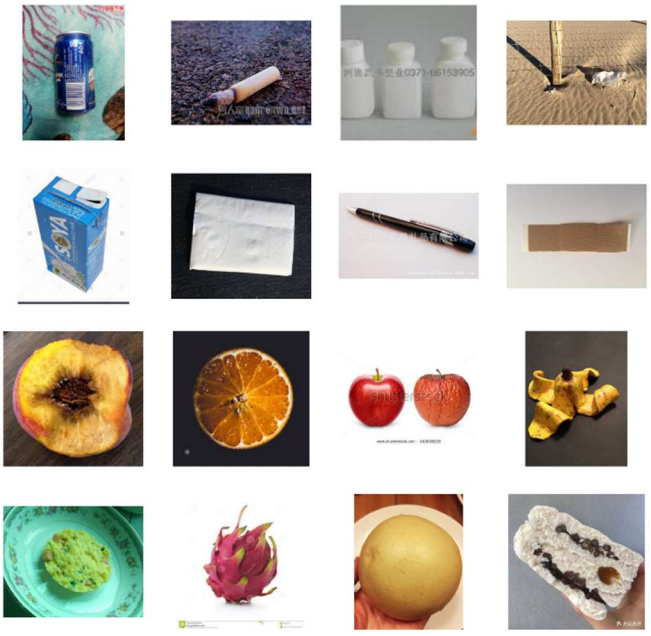

Fig. 1. Sample Data.

2.2. Single Shot Multibox Detector

SSD is a single-stage object detector architecture that utilizes feature maps with various scales and maps the default bounding box with various aspect ratio dimensions on each feature map to predict the score of each object label and the predicted bounding box offsets. Unlike other methods that require additional processing to generate bounding boxes, SSD defines default bounding boxes manually to shorten the detection time [2].

To support high quality detection, SSD requires an additional network called the base network and several new layers which decrease in size progressively. Unlike Multibox [5], in SSD, The default boxes used in training are selected manually and calculated to cover various real objects. For *m* feature maps, the default box scale used for each feature map is as follows.

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m-1}(k-1), \text{ where } k \in [1, m] \quad (1)$$

Where *Sk* is default box scale in feature map *k*, *Smin* is the scale of lowest layer in model, and *Smax* is the scale of highest layer in model. In each feature map cell, there is a default box that has a different aspect ratio, which are 1, 2, 3, ½ dan ⅓. Each default box has a width (*w*) and height (*h*) calculated as follows.

$$w = scale . \sqrt{aspect\ ratio} \quad (2)$$

$$h = \frac{scale}{\sqrt{aspect\ ratio}} \quad (3)$$

SSD also adds one more default box to the cell with aspect ratio = 1, which has the following scale.

$$s'_k = \sqrt{s_k . s_{k+1}} \quad (4)$$

SSD divides the prediction results into positive and negative. The default box will be marked as positive if it has a jaccard overlap of more than 0.5 with respect to the ground truth box, otherwise negative. Each default box will be paired with all ground truth boxes that have positive results, so that one ground truth box can have more than one default box.

After the results of matching the default box with the ground truth box have been obtained, the SSD calculates a loss function consisting of 2 components, namely localization loss (Lloc) and confidence loss (Lconf). The loss function is calculated as follows.

$$L(x, c, l, g) = \frac{1}{N} \left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \quad (5)$$

Where x is every positive default box, c is confidence of each class in the default box, l is predicted bounding box, g is ground truth box, N is number of positive default boxes, α is The weight of localization loss to the final loss value.

Confidence loss (Lconf) is calculated as follows.

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^{p} \log\left(\hat{c}_i^p\right) \sum_{i \in Neg} \log(\hat{c}_i^0) \quad (6)$$

Where $\hat{c}_i^p$ is the softmax loss of positive class confidence (p) against other classes detected by the default box (i).

Localization loss (Lloc) is calculated as follows.

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in cx,cy,w,h} x_{ij}^{p} smooth_{L1}\left(l_i^m - \hat{g}_j^m\right) \quad (7)$$

Where $\hat{g}_j^m$ is the difference in coordinates between the default box and the ground truth box.

And SSD also implements additional strategies, such as Hard Negative Mining to get rid of some bad results while training, Non Maximum Supression to limit the prediction of each object to the desired results, and Data Augmentation to give the model more perspective on various object sizes.

2.3. Modified VGG-16

VGG-16 is a convolutional neural network model with a substantial additional layer depth to the network. The total of layers in original VGG-16 is 13 convolution layers with 5 max-pooling layers in between and followed by 3 fully-connected layers [6]. In this paper, the base network of SSD is VGG-16 modified by using L2 normalization in conv4_3 layer, removing the fc8 layer and the dropout layers for fc6 and fc7 layers, converting fc6 and fc7 to convolutional layers, using the atrous convolution [7] in fc6 layers, and modifying the pool5 layer.

2.4. Real Time Detection

In this research, the measurement of detection time is frames per second (fps), which is calculated from how many images (frames) the system has detected in 1 second. The speed limit that must be passed by the system to be considered a real time detector is **24 fps**, which is the lowest frame rate a reguler camera has.

3. Methodology

The research approach in this study is shown in Fig. 2 below.

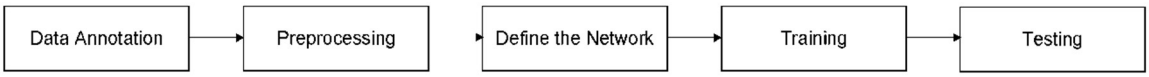

Fig. 2. Research Approach.

3.1. Data Annotation

The data annotation in this research is done with the help of makesense.ai by Piotr S. The objects in the image are annotated with bounding boxes and labels that represent the type of waste of the related object. The annotation format needed in this research is an XML file with Pascal VOC format [8]. After the annotation is given, there are some information we can conclude from the data as follows.

- The majority of images have dimensions of less than 1000×1000;

- The majority of images have normal aspect ratios between 0,5 and 2,5;
- There are more organic waste objects than inorganic waste objects in training data and testing data;
- There are too many small objects with a size scale of less than 0.1 in the images;
- Most of the ground truth boxes have an aspect ratios of less than 0,4.

3.2. Preprocessing

Before the data is used for model development, the data will go through several processes as follows.

a. Create Labelmap.

Labels are mapped in protocol buffer format as a label names reference for annotations stored in database, labelmap shown in Fig. 3.

```
item {
  name: "none_of_the_above"
  label: 0
  display_name: "background"
}
item {
  name: "organik"
  label: 1
  display_name: "organik"
}
item {
  name: "anorganik"
  label: 2
  display_name: "anorganik"
}
```

Fig. 3. Waste Type Labelmap.

b. Record Image Size Information.

Dimensional size information of each testing image data is recorded to readjust the bounding box size scale when documenting the model test results.

c. Store Images and Annotations in LMDB Database.

Each training data and testing data and their annotations are accessed in the LMDB key-value database.

3.3. Define The Network

In this research, there are 2 types of model networks built, namely SSD300 and SSD512. Each model has a difference in the input dimensions. SSD300 will process 300 × 300 image input, while SSD512 will process 512 × 512 image input. Both models developed with Caffe [9], so the training network and testing network of both models need to be define in protocol buffer format. The network specifications are as follows.

1) Training Network

a. Data Layer

The data layer is used to generate data blob and label blob by retrieving image data along with bounding box annotations which are mapped into the LMDB database, the data will be taken as much as the specified batch size in one iteration.

b. The Modified VGG-16 as specified in 2.3.

c. Additional Convolutional Layers.

Additional convolution layers on the SSD300 and SSD512 models are shown in the Table 1 and Table 2, respectively. For each existing convolution layer, all weight

parameters will be initialized using the xavier method [10] and bias parameters will be initialized with 0.

d. Priorbox Layer

Prior Box layer is the layer in charge of mapping the default box on the feature map. The priorbox layer retrieves the feature map of the layer that is used to generate predictions.

Table 1. Additional Layer on SSD300 Model

Layer Name	Type	Output	Kernel Size	Pad	Stride	Dilation
Conv 8_1	Convolution	256	1 × 1	0	1	1
Relu 8_1	ReLU	256	-	-	-	-
Conv 8_2	Convolution	512	3 × 3	1	2	1
Relu 8_2	ReLU	512	-	-	-	-
Conv 9_1	Convolution	128	1 × 1	0	1	1
Relu 9_1	ReLU	128	-	-	-	-
Conv 9_2	Convolution	256	3 × 3	1	2	1
Relu 9_2	ReLU	256	-	-	-	-
Conv 10_1	Convolution	128	1 × 1	0	1	1
Relu 10_1	ReLU	128	-	-	-	-
Conv 10_2	Convolution	256	3 × 3	0	1	1
Relu 10_2	ReLU	256	-	-	-	-
Conv 11_1	Convolution	128	1 × 1	0	1	1
Relu 11_1	ReLU	128	-	-	-	-
Conv 11_2	Convolution	256	3 × 3	0	1	1
Relu 11_2	ReLU	256	-	-	-	-

Table 2. Additional Layer on SSD512 Model

Layer Name	Type	Output	Kernel Size	Pad	Stride	Dilation
Conv 8_1	Convolution	256	1 × 1	0	1	1
Relu 8_1	ReLU	256	-	-	-	-
Conv 8_2	Convolution	512	3 × 3	1	2	1
Relu 8_2	ReLU	512	-	-	-	-
Conv 9_1	Convolution	128	1 × 1	0	1	1
Relu 9_1	ReLU	128	-	-	-	-
Conv 9_2	Convolution	256	3 × 3	1	2	1
Relu 9_2	ReLU	256	-	-	-	-
Conv 10_1	Convolution	128	1 × 1	0	1	1
Relu 10_1	ReLU	128	-	-	-	-
Conv 10_2	Convolution	256	3 × 3	1	2	1
Relu 10_2	ReLU	256	-	-	-	-
Conv 11_1	Convolution	128	1 × 1	0	1	1
Relu 11_1	ReLU	128	-	-	-	-
Conv 11_2	Convolution	256	3 × 3	1	2	1
Relu 11_2	ReLU	256	-	-	-	-
Conv 12_1	Convolution	128	1 × 1	0	1	1
Relu 12_1	ReLU	128	-	-	-	-
Conv 12_2	Convolution	256	4 × 4	1	1	1
Relu 12_2	ReLU	256	-	-	-	-

e. Localization Layer

This layer is used to predict localization by taking a feature map from each layer that is used to generate predictions. The localization layer produces many localization

predictions, so all the prediction results must be combined into one data in order to get the loss value. To simplify, the order of the dimensions of each blob will be changed from ($N \times C \times H \times W$) to ($N \times H \times W \times C$), then the data dimensions will be flattened so that they have the final dimensions ($N \times CHW$).

f. Confidence Layer

This layer also takes a feature map from each layer used to generate predictions and combines the results of confidence predictions into one data in the same way.

g. Loss Layer

Loss layer is a layer that assesses the accuracy of the prediction results by comparing it with the original data label, then calculates the loss value to calculate the gradient of parameter changes when backward processing.

2) Testing Network

The testing network structure is the same as the training network, but without the data layer and loss layer, and there are additional 2 layers at the end of the network, namely the detection output layer and the detection evaluate layer.

a. Detection Output Layer

This layer is used to produce the final detection results after getting the localization and confidence prediction results. The results of the localization prediction will map the default box to produce the final bounding box, then each bounding box will get its class based on the largest confidence value. The number of bounding boxes will be trimmed with NMS with a confidence threshold of 0.01 and a jaccard overlap of 0.45.

b. Detection Evaluate Layer

This layer is in charge of evaluating the accuracy of the final detection results on the label data. Each predicted bounding box will be classified as true positive or false positive with a jaccard overlap of 0.5.

3.4. Training

The models will be trained with a batch size of 32 for the SSD300 and SSD512 models, respectively. The SSD300 and SSD512 model training will be conducted with a momentum of 0.9 and an initial learning rate of 0.001. The training of the two models will last for 60,000 iterations with a decrease in learning rate of 0.1 in the 30,000 and 50,000 iterations.

3.5. Testing

The testing phase is carried out by evaluating the model on all testing data every 5000 iterations of training with the 11 Point Interpolated Mean Average Precision (mAP) metric. To get the mAP value, the Average Precision (AP) value for each class will be used, which will use prediction results that have confidence 0.5 and then will be sorted based on the greatest confidence. The prediction results are divided into the following.

a. True Positive (TP)

TP is the predicted bounding box that has a jaccard overlap of 0.5 or more to the ground truth box.

b. False Positive (FP)

FP is divided into 2 cases.

- If the predicted bounding box has a jaccard overlap of 0.5 but detects an object that has been predicted by the bounding box that has greater confidence.
- The predicted bounding box that has a jaccard overlap of less than 0.5 with respect to the ground truth box.

c. False Negative (FN)

FN is a ground truth box that has no predictive results.

Then each prediction result will calculate the value of precision and recall based on the accumulation of TP and FP from the top order. Precision and recall will be calculated in the following.

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

$$Recall = \frac{TP}{TP + FN} \quad (9)$$

By using precision and recall of each prediction result, AP can be calculated by finding the mean precision of 11 recall points {0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.8, 0.9, 1}. The precision value for each recall point is interpolated by taking the largest precision that has a recall greater than or equal to the current recall point.

$$\text{AP} = \frac{1}{11} \sum_{r \in \{0,0.1,..,1\}} \max_{\tilde{r} \geq r} \rho(\tilde{r}) \quad (10)$$

With $\rho(\tilde{r})$ is interpolated precision on current recall (r). After the AP value for each class (c) is obtained, the mAP can be calculated as follows.

$$mAP = \frac{\sum_c AP_c}{c} \quad (11)$$

4. Result and Discussion

4.1. Accuracy Test Result

During the training, the accuracy test occurred 12 times and the results of the SSD300 and SSD512 model accuracy testing can be seen in the Table 3.

Table 3. Accuracy Test Result

Test Phase	SSD300			SSD512		
	AP Organic	AP Inorganic	mAP	AP Organic	AP Inorganic	mAP
1	0,29698	0,227991	0,262485	0,51221	0,599599	0.555905
2	0,436507	0,427085	0,431796	0.587126	0.619711	0.603418
3	0,460086	0,434345	0,447216	0.590002	0.593967	0.591984
4	0,525609	0,483023	0,504316	0.595622	0.638512	0.617067
5	0,543945	0,512031	0,527988	0.528069	0.592964	0.560517
6	0,563468	0,521079	0,542273	0.60364	0.623703	0.613671
7	0,594694	0,542778	0,568736	0.614441	0.615085	0.614763
8	0,587322	0,543875	0,565598	0.628538	0.617908	0.623223
9	0,60271	0,552715	0,577712	0.633927	0.620482	0.627204
10	0,586917	0,546735	0,566826	0.623512	0.622308	0.62291
11	0,589812	0,548872	0,569342	0.620313	0.623755	0.622034
12	0,590261	0,550001	0,570131	0.64123	0.629746	0.635488

The cells colored in blue are the final performance of mAP and AP model accuracy.

4.2. Detection Speed Test Result

Testing the speed of the model is done by detecting 1000 testing data using the final model to be tested. The speed test is run with 2 configurations, detecting data in batches of 1 to see real time detection performance and detecting data in batches of 8 to see detection performance when utilizing hardware to its full potential. The results of the SSD300 and SSD512 model detection speed testing can be seen in the Table 4.

Table 4. Detection Speed Test Result

Model	Batch Size	Speed (*fps*)
SSD300	1	51
	8	78
SSD512	1	28
	8	36

5. Conclusion

Based on the tests carried out, the accuracy performance of the SSD300 and SSD512 final models are 0,57 mAP and 0,63 mAP, respectively, as can be seen in Table 3. It means that SSD512 has the better accuracy than SSD300, the decrease of image dimension in SSD300 causing the SSD300 model to be less sensitive to detecting features of small objects in the data than the SSD512 model.

And for the detection speed result in Table 4, the detection speeds that the SSD300 model and SSD512 model are capable of are 51 fps and 28 fps. This shows that the SSD300 model is able to detect faster than the SSD512, this is because the SSD300 model has less layer depth than the SSD512 model. Even so, both models are still able to achieve the detection speed needed to detect objects in real time.

References

[1] S. Bansal, S. Patel, I. Shah, A. Patel, J. Makwana, and R. Thakker, “AGDC: Automatic Garbage Detection and Collection”, arXiv preprint arXiv:1908.05849v1, 2019.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A.C. Berg, “SSD: Single Shot Multibox Detector”, arXiv preprint arXiv:1512.02325v5, 2016. (*main references*)

[3] P.F. Proenca and P. Simões, “TACO: Trash Annotations in Context for Litter Detection”, arXiv preprint arXiv: 2003.06975v2, 2020.

[4] J. Deng, W. Dong, R. Socher, L-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database”, IEEE conference on computer vision and pattern recognition 2009:248-255, 2009.

[5] D. Erhan, C. Szegedy, A. Toshev and D. Anguelov, “Scalable Object Detection using Deep Neural Networks”, arXiv preprint arXiv:1312.2249v1, 2013.

[6] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition”, 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc, 2015.

[7] L-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A.L. Yuille, “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution,and Fully Connected CRFs”, arXiv preprint arXiv:1606.00915v2, 2017.

[8] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn and A Zisserman, "The PASCAL Visual Object Classes Challenge: A Retrospective", International Journal of Computer Vision 111(1):98-136, 2015.

[9] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long , R. Girshick, S. Guadarrama, and T. Darrell, Caffe: Convolutional Architecture for Fast Feature Embedding”, arXiv preprint arXiv:1408.5093v1, 2014.

[10] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks”, In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS’10), 2010.