Faster RCNN based Garbage Detection System for Waste Management Education

Christian Teutsch
University of Heidelberg
Master Data and Computer Science
christian.teutsch@stud.uni-heidelberg.de

Josch Hagedorn
University of Heidelberg
Master Physics
josch.hagedorn@stud.uni-heidelberg.de

Ibrahima Sory Kourouma
University of Heidelberg
Master Computer Engineering
ibrahima.kourouma@stud.uni-heidelberg.de

Abstract

Waste management is a critical issue that affects both the environment and human health. Proper waste management is not only a responsibility of the local government, but also of each citizen. Often, the public's lack of awareness leads to improper waste separation and disposal. With the use of Machine Learning, especially Deep Learning, many novel applications have emerged. These also includes detectors that can identify different type of waste in an image or video. This work explores the garbage detection based on Faster RCNN and how it can be potentially used in a system that allows for educating people in waste management.

1. Introduction

The urbanisation across the globe continues to advance. For 2018 it was estimated by the United Nations that approximately 55% of the global population lived in towns or cities. That figure is expected to rise to 60% by 2030 [31,]. Due to this rapid urbanisation, the generation of municipal solid waste (MSW) is ever-increasing — even exceeding the growth rate of urbanisation itself. Waste generation in world cities was estimated at 1.3 billion tonnes in 2012 and is expected to almost double to 2.3 billion tonnes by 2025 [11,]. MSW consists of different things, such as food, paper and cardboard, glass, metals, plastics and textiles. [38,].

According to Hoornweg and Bhada-Tata, waste collection rates vary greatly between countries, correlating with income levels. Lower income countries are stated to have a collection rate of almost 50% in urban areas, middle-income countries have a rate varying from 71-85% and high income countries have collection rate of 100%. Often times, low-income countries do not have the additional financial resources to expand the waste management [11,].

However, waste management is not a task, that should be only addressed and taken on by the local government, but it starts already on the individual level. Two aspects can be considered when it comes to individual behaviour and responsibility. Firstly, it is important that the waste is disposed into containers and not dumped in unauthorized areas. In low- and middle-income countries, MSW is often dumped in low-lying area, on the road or into rivers [26, 11,]. Such a practice leads to contamination of groundwater and surface water [11,] Further more, insects and rodents attracted by waste can spread diseases such as cholera and dengue fever [1,].

Secondly, it is important to correctly separate the waste and dispose it into the designated containers. Incinerators and landfills are designed for specific categories of waste, making it difficult to directly incinerate or landfill mixed waste. When mixed waste is disposed, recyclable materials must be first recovered from this waste [24,]. Thus leading to an increase of down stream costs [17,].

One of the reasons for insufficient waste separation, is the lack of education regarding this matter [3,]. Different approaches exist to tackle this issue. For example through changes in the educational system by making it an integral part of the school curriculum [5,], or through technology, such as mobile applications to rise awareness [17,].

In the last two decades, machine learning (ML) has emerged as a sensible technology or method to tackle new problems. in different fields, such as computer vision [32,]. Machine learning in computer vision is often based on deep learning with Convolutional Neural Networks (CNN) used

for classifying objects into there respective classes [23,].

Different approaches exist for detecting objects in a frame, such as You Only Look Once (YOLO), Single Shot Detection (SSD) and Faster Residual Convolutional Neural Network (RCNN) [6,].

Numerous works already exist on the subject of object detection of garbage that make use of the aforementioned classification and detection methods, such as the work from Midday et al. [25,], Ma et al. [22,] and Wu [9,].

In this work, we focus on object classification and detection of garbage using Faster RCNN. The usage of different CNN backbones is explored. The goal is to find suitable detection models that can be used for developing a system that allows for holding an object up to the camera and receiving a suggestion for the type of waste. Potentially, this can be expanded to a system that also suggests the appropriate container into which the object must be disposed of. That can be easily achieved by using predefined rules. The first and most important step is to correctly detect and classify the object. Such a system deployed in households would lower the barrier to waste separation.

2. Theoretical Background

In this section a theoretical background about convolutional neural networks (CNNs) and object detection will be given.

2.1. CNNs

Multilayered perceptrons (MLPs) [29] are able to map input vectors $\mathbf{x} \in \mathbb{R}^D$ to outputs. Using a nonlinear activation function $\phi()$ and weights $\mathbf{W}$, at each hidden layer of the MLP activations $\mathbf{z} = \phi(\mathbf{W}\mathbf{x})$ are computed. If the input is chosen to be an image of width W, height H, and channel number C ($C = 3$ for RGB color), the input vector would be $\mathbf{x} \in \mathbb{R}^{WHC}$. Even for reasonably sized images the large number of parameters are a problem as the weight matrix would be of size $(W \times H \times C) \times D$, where D is the number of outputs. Another issue is that the model lacks translation invariance, meaning it may not recognize a pattern occurring in a new location if the weights are not shared across locations. In other words, a pattern that appears in one place may not be identified when it appears in a different location.

In order to address these issues, CNNs can be employed, where instead of matrix multiplication, a convolution operation is used. The approach involves breaking down the input into overlapping 2d image patches, and comparing each patch with a set of small weight matrices, referred to as filters, which represent different parts of an object. This is akin to a form of template matching, and the templates are learned from data. Due to their small size (typically 3x3 or 5x5), the number of parameters is considerably reduced. Additionally, using convolution for template matching instead of matrix multiplication ensures that the model

-	-	1	2	3	4	-	-	
7	6	5	-	-	-	-	-	$z_0 = x_0w_0 = 5$
-	7	6	5	-	-	-	-	$z_1 = x_0w_1 + x_1w_0 = 16$
-	-	7	6	5	-	-	-	$z_2 = x_0w_2 + x_1w_1 + x_2w_0 = 34$
-	-	-	7	6	5	-	-	$z_3 = x_1w_2 + x_2w_1 + x_3w_0 = 52$
-	-	-	-	7	6	5	-	$z_4 = x_2w_2 + x_3w_1 = 45$
-	-	-	-	-	7	6	5	$z_5 = x_3w_2 = 28$

Figure 1. Convolution of $\mathbf{x} = [1, 2, 3, 4]$ with $\mathbf{w} = [5, 6, 7]$, resulting in $\mathbf{z} = [5, 16, 34, 52, 45, 28]$ (image from [28])

is translationally invariant, making it useful for tasks like image classification, where the goal is to identify if an object is present, regardless of its location.

Given two functions $f, g : \mathbb{R}^D \to \mathbb{R}$, convolution is defined as

$$[f \circledast g](\mathbf{z}) = \int_{\mathbb{R}^D} f(\mathbf{u})g(\mathbf{z} - \mathbf{u})d\mathbf{u} \tag{1}$$

Applied to two vectors $\mathbf{w} = [w_{-L}, ..., w_L]$ and $\mathbf{x} = [x_{-N}, ..., x_N]$ equation 1 becomes

$$\begin{aligned}[\mathbf{w} \circledast \mathbf{x}](i) =& w_{-L}x_{i+L} + ... + w_{-1}x_{i+1} \\ &+ w_0x_i + w_1x_{i-1} + ... + w_Lx_{i-L}\end{aligned} \tag{2}$$

The process of applying the function 2 is shown in figure 1. If the weight vector w is symmetric, equation 2 simplifies to

$$[\mathbf{w} \circledast \mathbf{x}](i) = \sum_{u=0}^{L-1} w_u x_{i+u} \tag{3}$$

where $\mathbf{w} = [w_0, ..., w_{L-1}]$ and $\mathbf{x} = [x_0, ..., x_{N-1}]$. For 2d, equation 3 becomes

$$[\mathbf{W} \circledast \mathbf{X}](i, j) = \sum_{u=0}^{H-1} \sum_{v=0}^{W-1} w_{u,v} x_{i+u,j+v} \tag{4}$$

where $\mathbf{W}$ is a 2d filter of size $H \times W$. A visual example of 2d convolution is given in figure 2. 2d convolution is like matching a template with an image patch, where the output will be high if they are similar. If the template represents an edge, then convolving with it will highlight regions with edges of that orientation. Convolution can be seen as a feature detection process, and the resulting output is called a feature map. Valid convolution produces an output of size $(x_h - f_h + 1) \times (x_w - f_w + 1)$, where $f_h \times f_w$ is the filter size and $x_h \times x_w$ is the image size. Same convolution allows the output to have the same size as the input by using zero-padding. This adds a border of 0s to the image. Neighboring output pixels are similar in value because their inputs overlap. To reduce redundancy and speed up computation using strided convolution, every s'th input is skipped. Using zero padding of size p_h and p_w and strides of s_h and s_w, the output is then of size:

$$\left\lfloor \frac{x_h + 2p_h - f_h + s_h}{s_h} \right\rfloor \times \left\lfloor \frac{x_w + 2p_w - f_w + s_w}{s_w} \right\rfloor \tag{5}$$

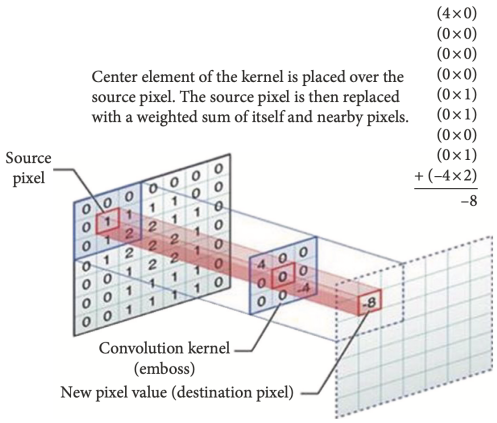

Figure 2. Example of a convolution (image from [2])

To handle inputs of multiple channels a kernel is defined for each input channel c, resulting in a 3d weight matrix $\mathbf{W}$. The output is computed as

$$z_{i,j} = b + \sum_{u=0}^{H-1} \sum_{v=0}^{W-1} \sum_{c=0}^{C-1} x_{si+u,sj+v,c} w_{u,v,c} \tag{6}$$

Here s is the stride and b is the bias term. This definition can be further extended by introducing multiple kinds of features d.

$$z_{i,j,d} = b_d + \sum_{u=0}^{H-1} \sum_{v=0}^{W-1} \sum_{c=0}^{C-1} x_{si+u,sj+v,c} w_{u,v,c,d} \tag{7}$$

Pointwise convolution allows changing the number of channels from C to D:

$$z_{i,j,d} = b_d + \sum_{c=0}^{C-1} x_{i,j,c} w_{0,0,c,d} \tag{8}$$

Pooling layers are a type of layer in a CNN that reduce the spatial dimensions (height and width) of the input volume, while retaining the depth dimension. This is achieved by applying a pooling function, such as max pooling or average pooling, to small local regions (also known as the receptive field) of the input. The pooling function outputs the maximum or average value of the elements in the local region, and the resulting output forms the corresponding output volume. By using pooling layers, the number of parameters in the network is reduced, and the network becomes more robust to variations in the input, such as translations or distortions. One commonly used approach in designing a CNN is to alternate between convolutional layers and max pooling layers, culminating in a linear classification layer at the end. This pattern is exemplified in figure 3. In this example a CNN is used to perform image classification. For this the function $f : \mathbb{R}^{H \times W \times K} \to \{0,1\}^C$ is learned, where K is

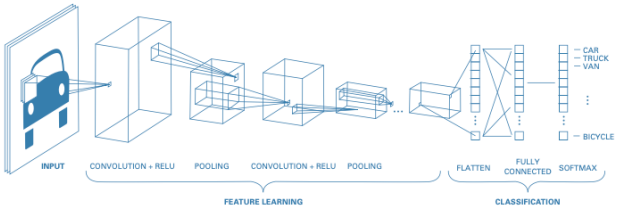

Figure 3. Simple CNN (image from [28])

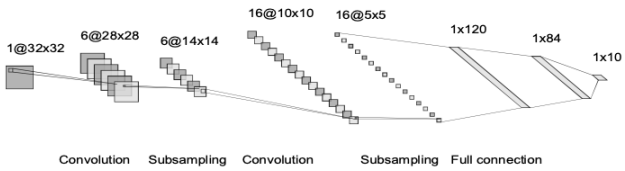

Figure 4. Architecture of LeNet-5 (image from [20])

the number of input channels and C is the number of class labels. [28]

In the following some common CNN architectures are presented.

LeNet-5 is a CNN designed by LeCun *et al.* [19] in 1998 for recognizing handwritten characters. It consists of seven trainable layers, including two convolutional layers, two pooling layers, and three fully-connected layers (see the architecture in figure 4). LeNet-5 combines local receptive fields, shared weights, and spatial or temporal sub-sampling to achieve shift, scale, and distortion invariance. However, it did not surpass traditional support vector machine (SVM) and boosting algorithms, and therefore did not receive much attention initially.

AlexNet is a deep CNN designed by Alex *et al.* in 2012 [18], which won the ImageNet 2012 competition. It has eight layers, including five convolutional layers and three fully-connected layers, and uses ReLU activation function, dropout, and local response normalization (LRN) for the first time on CNN (see architecture in figure 5). AlexNet also makes use of GPUs for computing acceleration. Its main innovations include using ReLU to mitigate the problem of gradient vanishing, dropout to avoid overfitting, overlapping max pooling to improve feature richness, LRN to enhance generalization ability, and two GPUs to train group convolutions. AlexNet also uses data augmentation to improve generalization ability.

He *et al.* proposed Residual Network (ResNet) in 2016 [10], which won the ILSVRC 2015 image classification and object detection algorithm. ResNet uses a two-layer or three-layer residual block constructed by shortcut connections to mitigate the gradient vanishing problem in deep neural networks (see architecture in figure 6). ResNet has inspired many studies to improve its performance, such

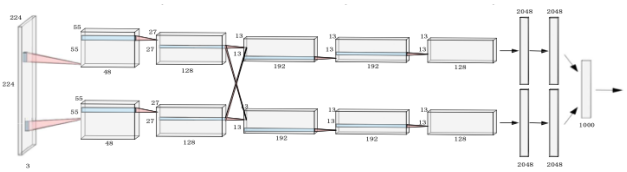

Figure 5. Architecture of AlexNet (image from [20])

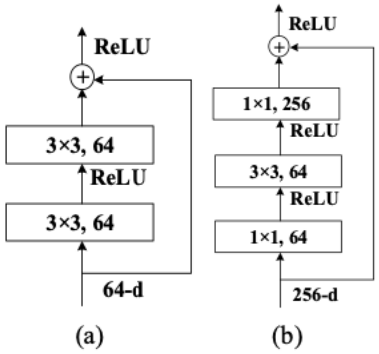

Figure 6. Architecture of (a) a two-layer residual block and (b) a three-layer residual block (image from [20])

as pre-activation ResNet, wide ResNet, stochastic depth ResNets, and ResNet in ResNet.

MobileNets are lightweight models designed by Google for embedded devices like mobile phones. There are three versions: MobileNet v1 [14], MobileNet v2 [39], and MobileNet v3 [12]. MobileNet v1 uses depth-wise separable convolutions and introduces width and resolution multipliers. MobileNet v2 adds inverted residual blocks and linear bottleneck modules, while MobileNet v3 uses platform-aware NAS and NetAdapt algorithm for network search, a lightweight attention model based on squeeze and excitation, and h-swish activation function. [20]

2.2. Object Detection

Object detection is a fundamental task in computer vision that aims to detect instances of objects in digital images. It is essential for many computer vision applications and is based on accuracy and speed metrics. Deep learning techniques have greatly improved object detection in recent years. Challenges in object detection include object rotation, scale changes, accurate localization, dense and occluded object detection, and speed. Figure 7 shows that object detection is still an active field of research with a growing number of publications per year. [44]

Object detection requires the identification of objects of interest and returning a collection of bounding boxes that represent their location and class labels. Face detection is a special instance of object detection that focuses on detecting a single class of object. To solve detection issues

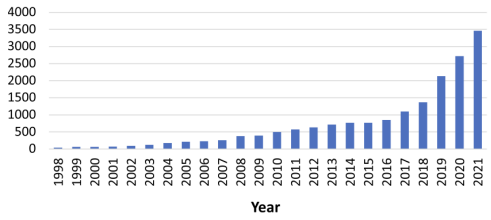

Figure 7. Number of publications in object detection (image from [44])

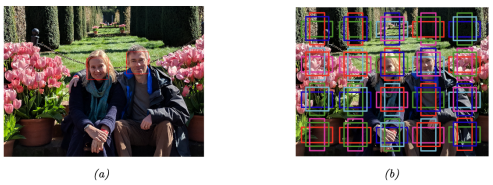

Figure 8. (a) Face detection and (b) Anchor boxes (image from [28])

like this, a trivial approach is to transform them into closed world problems, in which there are a limited number of potential object locations and orientations, known as anchor boxes. The system is taught to forecast which object category, if any, is present within each box, and may also carry out regression to anticipate the deviation of the object's location from the anchor's center. In figure 8 bounding boxes for the purpose of face detection and an illustration of anchor boxes is shown. The process of object detection using anchor boxes can be described by the function

$$f_\theta : \mathbb{R}^{H\times W\times K} \rightarrow [0,1]^{A\times A} \times \{1,...,C\}^{A\times A} \times (\mathbb{R}^4)^{A\times A} \tag{9}$$

with input channels K, anchor boxes in each dimension A, and class labels C. The predictions for each box location (i, j) are an object presence probability, $p_{ij} \in [0, 1]$, an object category, $y_{ij} \in \{1, ..., C\}$, and two 2d offset vectors, $\delta_{ij} \in \mathbb{R}^4$. [28]

During the traditional object detection period, before 2014, object detection algorithms relied on handcrafted features due to the lack of effective image representation. Some of the milestone detectors during this period include the Viola Jones detector, the HOG detector, and the Deformable Part-Based Model (DPM). These detectors were able to achieve real-time detection of human faces and other objects with high accuracy by using sliding windows, feature selection, detection cascades, and other techniques. Although the accuracy of these detectors has been surpassed by deep learning-based detectors, they have laid the foundation for many object detectors and computer vision applications.

VOC 2010 test	mAP
$DPMv5$	33.4
UVA	35.1
$Regionlets$	39.7
$SegDPM$	40.4
$R-CNN$	50.2
$R-CNNBB$	53.7

Table 1. Detection average precision on VOC 2010 test.

The deep learning-based detection period began with the rebirth of CNNs in 2012, which can learn robust and high-level feature representations of an image. The first breakthrough during this period was the Regions with CNN features (RCNNs) in 2014. RCNNs were able to achieve state-of-the-art object detection accuracy by using a two-stage detection process, which involved extracting object proposals using selective search and feeding them into a CNN model to extract features. Other two-stage detectors that followed include Fast R-CNN, Faster R-CNN, and Mask R-CNN.

One-stage detectors were also introduced during this period, which aimed to complete the detection process in one step. Some of the popular one-stage detectors include YOLO (You Only Look Once) and SSD (Single Shot Detector). These detectors have been able to achieve real-time object detection on low-power devices and have been widely used in various applications. [20]

2.3. Detectors

Detectors combine the classification of an object with its localization in the image. A classification model provides only information if it classified a trained class inside an image. However, the user hasn't any feedback if the model focuses on the right object in the image. Detectors are a modification of a classifier that additionally draws a box around the object of interest. There exist different architectures of detectors. We will focus on Faster R-CNN as it was introduced in the lecture and pre-trained networks are accessible. In addition, we show the development of Faster R-CNN by explaining the previous architectures of R-CNN and Fast R-CNN. In the end, we compare the R-CNN architectures with another type of detector called YOLO.

2.4. R-CNN

In 2014 Girshick et. al published a paper that introduced new detection approach by combining convolutional neural networks (CNNs) with region proposal methods. The new architecture outperformed the state-of-the-art algorithms as shown in Table 1 [8].

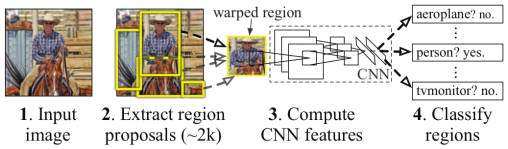

Figure 9. R-CNN: Regions with CNN features

2.4.1 Architecture

The architecture of R-CNN is based on three modules shown in Figure 9. The first module is the region proposal method. One of the several existing region proposals is selective search [40] and is used in R-CNN. Selective search proposes around 2000 regions in images that are likely to contain an object. The second module is a CNN which extracts feature vectors from each region proposal. The outcoming feature vector is classified by a linear support vector machine (SVM). The SVM is the third module of the R-CNN architecture [8].

2.4.2 Training

The CNN, used for feature extraction, was pre-trained by Girshick et al. on the ILSVRC 2012 dataset. To train the network on the VOC dataset Girshick et al. reduced the classification layer from 1000 ILSVRC-classes to 20 VOC-classes plus a background class. The R-CNN is trained with stochastic gradient descent (SGD). All region proposals with an overlap higher than 0.5 to the ground truth are count as positive [8].

2.4.3 Drawbacks

Despite the state-of-the-art precision in the year 2015, R-CNN has worth mentioning drawbacks in training and detection time. The long training time is attributed to the multi-stage learning. The modules of the architecture are trained sequentially. Due to the individual classification of each region proposal, R-CNN needs around 13s/image on a GPU to detect an object in an image [8].

2.5. Fast R-CNN

In 2015 Girshick introduced the devolopment of R-CNN called Fast R-CNN. The new architecture takes advantage of shared computation. The improved version beats its previous architecture by a 9 x faster training time, a 213 x faster test time, and higher mean average precision (mAP) on PASCAL VOC [7].

2.5.1 Architecture

Instead of extracting a feature vector for every single region proposal Fast R-CNN extract a feature map of the whole in-

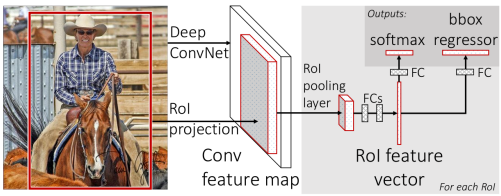

Figure 10. R-CNN: Regions with CNN features

put image shown in Figure 10. In the following module, a region of interest (RoI) pooling layer extracts the information from the feature map which are the region proposals. The region proposal vector is traversed into a fully connected layer. The last module consists of a classification softmax and a bounding box regressor which are fed by the splitted fully connected layers [7].

2.5.2 Training

For the feature map extraction, Girshick used different pretrained networks which has to be modified such that the max pooling layer is replaced by the introduced RoI pooling layer, the classification layer is replaced with the classification softmax and the bounding box regressor and an additional input channel for the list of RoIs of the training images [7].

The training is optimized by feature sharing which is realized by a hierarchical sampling approach. Inside a batch, the algorithm samples N images and extracts N/R RoIs from each image. The features of the RoIs from the same image can be shared which is an improvement to R-CNN where each RoI comes from a different image. Fast R-CNN optimizes the weights with SGD using back-propagation. As mentioned before R-CNN uses a time-consuming multi-stage approach during the training. Fast R-CNN trains its weights in only one stage by optimizing the softmax classifier parallel to the bounding-box regressor [7].

2.6. Faster R-CNN

The architecture used for garbage detection is called Faster R-CNN. The bottleneck of the previous Fast R-CNN consists in the computation time of the region proposals which needs much as time as the detection network. In 2017 Ren et al. introduced a Region Proposal Network (RPN) to reduce computation costs by sharing the convolution layers with object detection networks [36].

2.6.1 Architecture

Faster R-CNN extends Fast R-CNN with an RPN from Figure 11. The RPN is used to extract RoI which is processed with the Fast R-CNN.

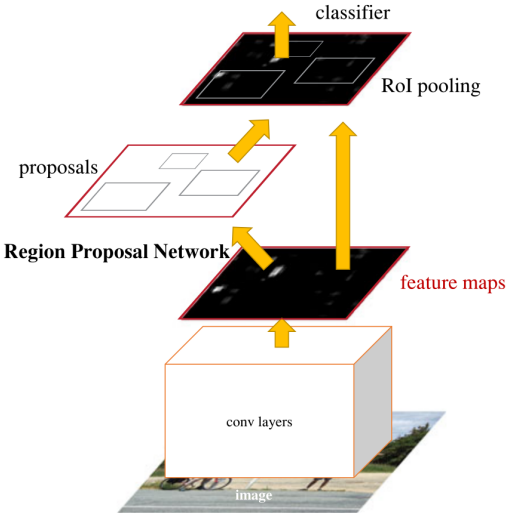

Figure 11. Faster R-CNN architecture

2.6.2 Region Proposal Networks

An RPN is modeled by a fully convolutional network. The region proposed by a small network consists of an n x n convolution layer branched into two 1 x 1 convolution layers. These convolutional layers are used for classification and bounding box regression. The small network slides over the resulting feature map of the CNN and extract the special type of region proposals which Ren et al. called anchors. At each position of the small network on the feature map proposed k different scaled anchors as a possible region of an object. The use of multi-scale anchors make it possible to share the features of the RPN with the features of the Fast R-CNN. This is a key improvement of Faster R-CNN.

2.6.3 Training

The RPN is trained with SGD using backpropagation. The training data batch comes from one image with 128 negative and 128 positive anchors. Faster R-CNN shares the convolutional layers for RPN and Fast R-CNN by using alternating training. The iterative approach uses the ouptut of the RPN as the input of the Fast R-CNN and vice versa.

2.7. YOLO

The R-CNN family belongs to the classification-based algorithms. To give a comparison with other detector classes we present YOLO (You Only Look Ones) which belongs to the class of regression-based algorithms. YOLO was introduced in 2016 and outperformed it competitors in the speed of detection which is shown in Table 2.

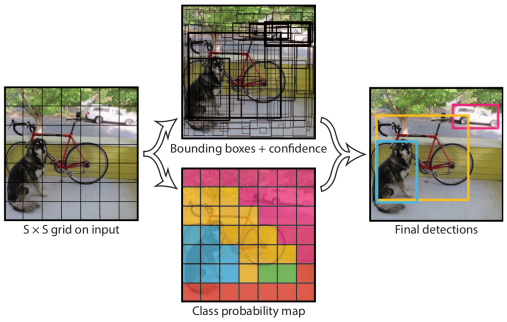

Figure 12. YOLO architecture

2.7.1 Unified Detection

The R-CNN detection algorithm classify propose regions with a high probabilitiy of containing an object. However, YOLO combines the detection in a single regression problem which can be tackled with a single CNN. The simplicity of this algorithm enables real-time processing [35].

YOLO splits the input image into a defined number of grid cells shown in Figure 12. The algorithm predicts a defined number of bounding boxes for each cell. Each bounding box is defined by 5 predictions which are the 2d center position of the box inside the grid cell and the width and height. In addition, the fifth prediction is the confidence score defined by the intersection over union (IOU). Another attribute of the grid cells containing an object is the conditional class probability. Together, the IOU and the conditional class probability lead to the class-specific confidence scores. These scores are used to classify a box by the probability of appearance and the fitting of the bounding box [35].

2.7.2 Architecture

Redmon et al. designed the architecture with a GoogLeNet-inspired CNN with 24 convolutional layers. The convolutional layers are followed by two fully connected layers for the prediction of the output probabilities and coordinates. The final predictions of the network are outputed by an 7x7x30 tensor at the end of the architecture. [35].

2.7.3 Training

The first 20 convolutional layers of YOLO architecture are pre-trained on the ImageNet 1000-class competition dataset. The other four convolutional layers are randomly initialized and added to the pre-trained convolutional layers. This method was accepted from [37] to improve the performance. The objective for optimization is a sum-squared error which has the drawback that localization and classification errors are equally weighted and lowers the mAP. To improve the training the optimization takes only the highest IOU box for each cell into account [35].

Detectors	mAP	FPS
$R-CNN\ Minus\ R$	53.5	6
$Fast\ R-CNN$	70.0	0.5
$Faster\ R-CNN\ VGG-16$	73.2	7
$Faster\ R-CNN\ ZF$	62.1	18
$YOLO\ VGG-16$	66.4	21
$Fast\ YOLO$	52.7	155
$YOLO$	63.4	45

Table 2. FPS and mAP trained on PASCAL VOC 2007

2.8. Detector Comparison

Table 2 compares the performance of the R-CNN family detectors with three different YOLO architectures. Fast YOLO beats all other detectors with an frames per second (FPS) of 155 in speed. However, the fast detection time contributes to a low mAP of 52.7. The standard YOLO architecture can be classified as a compromise in speed and precision. With an FPS of 45, the model is able to detect objects in real-time. The precision with an mAP of 63.4 is higher than Fast YOLO but lower than F aster R-CNN VGG-16 with an mAP of 73.2.

3. State of the art

Numerous previous works exist for both classifying and detecting garbage. We have summarized some of these works below. However, it's worth noting that state of the art methods for detecting garbage, which mainly focus on identifying trash in busy frames, are only somewhat relevant to our work. Nonetheless, we've included them below for the sake of completeness.

3.1. Garbage classification

Kang *et al*. proposed an automatic garbage classification system based on deep learning in 2020. [16] The work includes a design for a novel garbage bin that can separate garbage into classes, recyclable and non-recyclable. The garbage classification algorithm that has been suggested utilizes the ResNet-34 algorithm as its foundation, but it has been enhanced in three key ways to optimize its network structure. These enhancements include multi-feature fusion of input images, the reuse of features within residual units, and the introduction of a novel activation function. The algorithm is trained with a total of 4168 pictures of 14 items. Those items are books, bottles, towels, paper boxes, batteries, banana skins, leaves, and orange skins. As per the results of the experiments conducted, the classification accuracy of the system is as high as 99%, and the classification cycle is as swift as 0.95 seconds.

Model	Softmax accuracy [%]	SVM accuracy [%]
AlexNet	87.14	97.23
GoogleNet	88.10	97.86
ResNet	89.38	94.22
VGG-16	90	97.46
SqueezeNet	80.43	90.17

Table 3. Results of the models after 200 epochs

Rabano *et al*. implemented a common garbage classification using MobileNet in 2018. [34] A MobileNet model was trained to classify six categories of common trash (glass, paper, cardboard, plastic, metal, and other trash) using a dataset of 2527 images. The model achieved a final test accuracy of 87.2%, which was optimized and quantized for mobile app development. The optimized model outperformed the quantized model, with 89.34% confidence versus 1.47% confidence, in a test using a plastic image. The model app was successfully installed and tested on a Samsung Galaxy S6 Edge+ mobile phone, successfully identifying a cardboard material in an image of a cardboard container. The study suggests retraining the model using more steps to improve the performance of the quantized model, which is suitable for mobile devices.

Özkaya and Seyfi fine-tuned and compared different models with the purpose of classifying garbage in 2019. [30] The used data set consist out of 2527 images of six different classes. Half of these images were used for training and the other half for testing. The five models AlexNet, GoogleNet, ResNet, VGG-16, and SqueezeNet were tested with both Softmax and Support Vector Machines (SVM) as classifiers. The results are summarized in table 3.

3.2. Garbage detection

Wang and Zhang implemented an autonomous garbage detection in 2018. [41] The study focuses on automatic garbage detection using a Faster R-CNN open source framework with a region proposal network and ResNet network algorithm. To enhance accuracy, a data fusion and augmentation strategy is proposed. The results of the experiments demonstrate that the method has a high-precision detection function and favorable generalization ability. The study emphasizes the importance of garbage detection in the application of intelligent urban management. The ResNet network is applied to the Faster R-CNN framework, which avoids gradient disappearance or explosion and precision degradation. The Faster R-CNN object detection method includes region proposal generation, feature extraction, classification, and location optimization. During training, the dataset image is inputted, the ResNet network generates the feature map, and the RPN layer generates a large number of region proposals. The architecture is shown in figure 13.

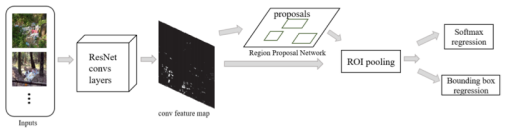

Figure 13. State of the art garbage detector (image from [41])

Figure 14. Qualitative performance of the state of the art method (image from [41])

The used dataset contained 816 images, with 596 used for training and 220 for testing. The images were taken from different scenes and directions to increase the diversity of the dataset. A data fusion strategy was used, and each background class had at least 50 images. Backgrounds include buildings, roads, and lawns. The qualitative performance of the approach is shown in figure 14.

De Carolis *et al*. developed a software in 2020 that can detect and report abandoned waste in real-time using a YOLOv3 network model, which has been fine-tuned with a dataset collected for this purpose. [4] The dataset used in the study, which includes 2714 images of objects belonging to four different classes (Garbage Bag, Garbage Dumpster, Garbage Bin, and Blob) and 1260 images of negative examples depicting landscapes without objects of interest. In total, there are 5535 labels in the dataset. The used YOLOv3 is a CNN consisting of 106 layers, with the first 53 layers serving as a pre-trained feature extractor on Imagenet. The remaining 53 layers are used for object detection on three different scales of objects. YOLOv3 also uses the anchor box method, which helps in training the network faster and more efficiently. In the study, the authors train the last 53 layers of YOLOv3 on their dataset and name it YOLO TrashNet. The qualitative performance of the approach is shown in figure 15.

Wu *et al*. proposed a garbage detection and classification method in 2021 based on visual scene understanding using knowledge graphs to store and model items in the scene. [42] The method combines the ESA attention mechanism with the YOLOv5 network to improve feature extraction and form the YOLOv5-Attention-KG model. Test results show that the proposed model has higher accuracy and real-time performance than the original YOLOv5 model and can realize intelligent decision-making for garbage classifica-

Figure 15. Qualitative performance of YOLO TrashNet (image from [4])

tion in a complex system.

4. Methodology

4.1. Data Description

The data used for this work have been obtained from Kaggle. Kaggle is a platform for exploring, creating and sharing data-related projects. It hosts a wide range of datasets [33, 15,].

The dataset from [27,] is mostly taken over — only the class trash is removed because the images are insufficient and not truly representing the class. It is used for training and assessing the CNNs Table 4.1. A dummy class background is added to the dataset, so that the CNN can be reused later as a custom backbone for the Faster RCNN,model since the library PyTorch implicitly adds the class background and the number of output features of the backbone needs to match with the RPN of the Faster RCNN.

A second dataset, which is a subset of the other, is used for training and validating the detection models. The images were carefully selected so that they allow for drawing bounding boxes that contain more or less only the object and not so much of the background Table 4.1.

Class	Number of images
Background	1
Battery	945
Biological	985
Brown-glass	607
Cardboard	891
Clothes	5325
Green-glass	629
Metal	769
Paper	1050
Plastic	865
Shoes	1977
White-glass	775

Table 4. Data set for garbage classification. In total 14819 images are used.

Class	Number of images	Number of instances
Battery	100	222
Biological	100	186
Brown-glass	100	216
Cardboard	100	185
Clothes	100	100
Green-glass	100	177
Metal	100	133
Paper	100	120
Plastic	100	113
Shoes	100	129
White-glass	100	132

Table 5. Data set for garbage detection

4.2. Model building

For testing the classification models, we use the pre-trained models ResNet50, MobileNetV3, EfficientNet and GoogLeNet. which are provided by PyTorch directly. Before training, all layers of the CNN of these models are frozen and the classification layer is replaced with a linear layer, so that only the weights of the linear layer are updated. This approach speeds up the training significantly.

Based on the results of the classification test, 2 CNN models are chosen to be used as backbones for 2 Faster RCNN models, They are compared to Faster RCNN models that are pre-trained on the CoCo dataset which contains 1.5 million images [21,].

4.3. Evaluation Metrics

Regarding the classification test, the proportion of correct predictions to total predictions is considered as the defining metric for comparing the models. For evaluating the Faster RCNN models, the mean average precision is used. This metric is commonly used for evaluating detection models [43,].

Data set size	Number of images
Small	2500
Medium	7500
Large	14819

Table 6. Data set sizes for classification test

Data set size	Number of images
Small	200
Medium	600
Large	1100

Table 7. Data set sizes for detection test

4.4. Training and validation

In each of the two tests, the models are trained on different sizes of training data (see Table 6, Table 7) The validation always takes place with the total data set. The RTX 3090 TI with 10752 cuda cores and 24 GB VRAM is used for running the benchmark test.

5. Results

Before the models are evaluated for their performance, they are first trained for 10 epochs. As can be seen in Table 8, regardless of the data set size, the biggest difference between two models, is approximately 2%, which means they are similar in performance. The small differences can possibly be explained by the fact that the training data is randomly shuffled for each test case.

This observation is, due to the small sample size, not statistically relevant and does not allow for drawing conclusions about the over all performance of these models.

Model	Dataset	Accuracy(%)
MobileNetV3	small	85.04
MobileNetV3	medium	88.31
MobileNetV3	large	89.45
Resnet50	small	86.61
Resnet50	medium	90.09
Resnet50	large	91.65
GoogLeNet	small	84.09
GoogLeNet	medium	88.22
GoogLeNet	large	89.23
EfficientNet	small	86.93
EfficientNet	medium	90.07
EfficientNet	large	91.2

Table 8. Results for classification test

As there is not much difference between the models, it was decided to use the ResNet50 and the MobileNetV3 model from the classification test to build the custom Faster RCNN models which are then compared in the detection test to the respective models of the same architecture provided by PyTorch.

Table 9 and Table 10 show that the models trained with the Coco dataset perform much better, than the other models which is not surprising, since this dataset consists of 1.5 million datasets. The MobileNetV3 based Faster RCNN shows better performance than the Resnet50 based one in all metrics.

Surprisingly, the Faster RCNN model with the custom MobileNetV3 backbone shows reasonable performance even though it was trained with a very small dataset.

Model (backbone)	Dataset	mAP[.5,.95](%)
MobileNetV3	small	54.40
MobileNetV3	medium	79.5
MobileNetV3	large	89.60
Resnet50	small	48.00
Resnet50	medium	65.60
Resnet50	large	84.40
Custom MobileNetV3	small	6.60
Custom MobileNetV3	medium	21.20
Custom MobileNetV3	large	31.50
Custom Resnet50	small	0.00
Custom Resnet50	medium	0.60
Custom Resnet50	large	1.40

Table 9. Results for detection test

Model (backbone)	Dataset	mAP.5(%)	mAP.75(%)
MobileNetV3	small	69.10	63.30
MobileNetV3	medium	92.90	89.0
MobileNetV3	large	98.40	97.0
Resnet50	small	61.40	57.60
Resnet50	medium	79.70	75.20
Resnet50	large	95.10	93.30
Custom MobileNetV3	small	15.90	3.70
Custom MobileNetV3	medium	45,80	14.70
Custom MobileNetV3	large	61.00	29.0
Custom Resnet50	small	0.00	0.00
Custom Resnet50	medium	3.20	0.00
Custom Resnet50	large	6.40	0.20

Table 10. Results for detection test

6. Discussion

Based on the results, it was decided that for the development of the garbage detection system for waste management education, a FasterRCNN model with a MobileNetV3 backbone is used. Not only does it show good results with limited amount of training data and time, it has been purposefully designed for considering platforms with lower computational resources which is important for such an application [13,].

However, when evaluating the model with a webcam feed, it was found that the performance of the image detection did not translate well to the detection of real objects. It is possible that the images in the data set lack of certain qualities that are required for the model to recognise real objects.

7. Conclusion

This work shows that the Faster RCNN based on MobileNetV3 is a model that is sensible to use in a system for waste management eduction. The efficiency of this model,

both in terms of computational resource requirements and training data, can be used as an advantage here.

In future work, it needs to be researched further how to better translate the performance of object detection in images to the detection of objects captured through a webcam or mobile cam.

References

[1] Pervez Alam and K. Ahmade. Impact of solid waste on health and the environment. *International Journal of Sustainable Development and Green Economics (IJSDGE)*, 2:165–168, 01 2013. 1

[2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017. 3

[3] Vera Basiroen, Mita Purbasari, and Elizabeth Yuliana. Prajurit alam: Introduction to waste separation education for first and second grade elementary students. *IOP Conference Series: Earth and Environmental Science*, 794:012119, 07 2021. 1

[4] Berardina De Carolis, Francesco Ladogana, and Nicola Macchiarulo. Yolo trashnet: Garbage detection in video streams. In *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–7. IEEE, 2020. 8, 9

[5] Justice Kofi Debrah, Diogo Guedes Vidal, and Maria Alzira Pimenta Dinis. Raising awareness on solid waste management through formal education for sustainability: A developing countries evidence review. *Recycling*, 6(1):6–0, 2021. 1

[6] R Deepa, E Tamilselvan, E.S. Abrar, and Shrinivas Sampath. Comparison of yolo, ssd, faster rcnn for real time tennis ball tracking for action decision networks. In *2019 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, pages 1–4, 2019. 2

[7] Ross Girshick. Fast R-CNN. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:1440–1448, 2015. 5, 6

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 5

[9] Wu Han. A yolov3 system for garbage detection based on mobilenetv3lite as backbone. In *2021 International Conference on Electronics, Circuits and Information Engineering (ECIE)*, pages 254–258, 2021. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[11] Dan Hoornweg and Perinaz Bhada-Tata. What a waste: a global review of solid waste management. *Urban Dev Ser Knowl Pap*, 15:87–88, 01 2012. 1

[12] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 4

[13] Andrew G. Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 10

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 4

[15] Yen-Hung Frank Hu, Abdinur Ali, Chung-Chu George Hsieh, and Aurelia Williams. Machine learning techniques for classifying malicious api calls and n-grams in kaggle data-set. In *2019 SoutheastCon*, pages 1–8, 2019. 9

[16] Zhuang Kang, Jie Yang, Guilan Li, and Zeyi Zhang. An automatic garbage classification system based on deep learning. *IEEE Access*, 8:140019–140029, 2020. 7

[17] S. Kaza, L.C. Yao, and P. Bhada-Tata. *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050*. Urban development series. World Bank Publications, 2018. 1

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 3

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3

[20] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 2021. 3, 4, 5

[21] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. 9

[22] Wen Ma, Xiao Wang, and Jiong Yu. A lightweight feature fusion single shot multibox detector for garbage detection. *IEEE Access*, 8:188577–188586, 2020. 2

[23] Supriya V. Mahadevkar, Bharti Khemani, Shruti Patil, Ketan Kotecha, Deepali R. Vora, Ajith Abraham, and Lubna Abdelkareim Gabralla. A review on machine learning styles in computer vision—techniques and future directions. *IEEE Access*, 10:107293–107329, 2022. 2

[24] Shigeru Matsumoto. Waste separation at home: Are japanese municipal curbside recycling policies efficient? *Resources, Conservation and Recycling*, 55(3):325–334, 2011. 1

[25] Asif Iqbal Middya, Debjani Chattopadhyay, and Sarbani Roy. Garbage detection and classification using faster-rcnn with inception-v2. In *2021 IEEE 18th India Council International Conference (INDICON)*, pages 1–6, 2021. 2

[26] Florin Mihai, Liviu Apostol, Adrian Ursu, and Pavel Ichim. Vulnerability of mountain rivers to waste dumping from

neamt county. *Geographia Napocensis*, 6:51–59, 11 2012. 1

[27] Mostafa Mohamed. Garbage classification (12 classes), 2021. 9

[28] Kevin P Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022. 2, 3, 4

[29] Fionn Murtagh. Multilayer perceptrons for classification and regression. *Neurocomputing*, 2(5-6):183–197, 1991. 2

[30] Umut Ozkaya and Levent Seyfi. Fine-tuning models comparisons on garbage classification for recyclability. *arXiv preprint arXiv:1908.04393*, 2019. 8

[31] United Nations Publications. *World Urbanization Prospects: The 2018 Revision*. UN, 2019. 1

[32] Raffaele Pugliese, Stefano Regondi, and Riccardo Marini. Machine learning-based approach: global trends, research directions, and regulatory standpoints. *Data Science and Management*, 4:19–29, 2021. 1

[33] Luigi Quaranta, Fabio Calefato, and Filippo Lanubile. Kgtorrent: A dataset of python jupyter notebooks from kaggle. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*, pages 550–554, 2021. 9

[34] Stephenn L Rabano, Melvin K Cabatuan, Edwin Sybingco, Elmer P Dadios, and Edwin J Calilung. Common garbage classification using mobilenet. In *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, pages 1–4. IEEE, 2018. 8

[35] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 7

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 6

[37] Shaoqing Ren, Kaiming He, Ross Girshick, Xiangyu Zhang, and Jian Sun. Object detection networks on convolutional feature maps. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1476–1481, 2016. 7

[38] Lesley Rushton. Health hazards and waste management. *Br Med Bull*, 68:183–197, 2003. 1

[39] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 4

[40] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104:154–171, 2013. 5

[41] Ying Wang and Xu Zhang. Autonomous garbage detection for intelligent urban management. In *MATEC Web of Conferences*, volume 232, page 01056. EDP Sciences, 2018. 8

[42] Yuezhong Wu, Xuehao Shen, Qiang Liu, Falong Xiao, and Changyun Li. A garbage detection and classification method based on visual scene understanding in the home environment. *Complexity*, 2021:1–14, 2021. 8

[43] Ming Zeng, Jiazhe Liu, Xiangzhe Lu, Zhijing Wang, Yuanhao Li, and Siying Li. An improved object detection algorithm for assembly line garbage sorting. In *2022 41st Chinese Control Conference (CCC)*, pages 7118–7123, 2022. 9

[44] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023. 4