applied sciences

MDPI

Article

Cascaded Machine-Learning Technique for Debris Classification in Floor-Cleaning Robot Application

Balakrishnan Ramalingam [1], Anirudh Krishna Lakshmanan [1,2], Muhammad Ilyas [1,3], Anh Vu Le [1,4] and Mohan Rajesh Elara [1,*]

1 Engineering Product Development Pillar, Singapore University of Technology and Design (SUTD), Singapore 487372, Singapore; balakrishnan_ramalingam@sutd.edu.sg (B.R.); anikl9705@gmail.com (A.K.L.); muhammad_ilyas@sutd.edu.sg (M.I.); anhvu_le@sutd.edu.sg (A.V.L.)
2 Department of Computer Science, Birla Institute of Technology and Science (BITS) Pilani, Pilani Campus, Vidyavihar, Rajasthan 333031, India
3 Department of Electrical Engineering, UET, NWL Campus, Lahore 54890, Pakistan
4 Optoelectronics Research Group, Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City 70000, Vietnam
* Correspondence: rajeshelara@sutd.edu.sg

Received: 5 November 2018; Accepted: 12 December 2018; Published: 17 December 2018

Abstract: Debris detection and classification is an essential function for autonomous floor-cleaning robots. It enables floor-cleaning robots to identify and avoid hard-to-clean debris, specifically large liquid spillage debris. This paper proposes a debris-detection and classification scheme for an autonomous floor-cleaning robot using a deep Convolutional Neural Network (CNN) and Support Vector Machine (SVM) cascaded technique. The SSD (Single-Shot MultiBox Detector) MobileNet CNN architecture is used for classifying the solid and liquid spill debris on the floor through the captured image. Then, the SVM model is employed for binary classification of liquid spillage regions based on size, which helps floor-cleaning devices to identify the larger liquid spillage debris regions, considered as hard-to-clean debris in this work. The experimental results prove that the proposed technique can efficiently detect and classify the debris on the floor and achieves 95.5% percent classification accuracy. The cascaded approach takes approximately 71 milliseconds for the entire process of debris detection and classification, which implies that the proposed technique is suitable for deploying in real-time selective floor-cleaning applications.

Keywords: Convolutional Neural Network (CNN); Support Vector Machine (SVM); floor-cleaning robot; object detection; debris classification; SSD MobileNet

1. Introduction

Floor-cleaning robots are widely used in food courts, hospitals, industries, and homes for picking up debris (dust, dirt, liquid spillage debris) and mopping floors. The critical challenge of floor-cleaning robots is to determine the type of debris and recognize easy-to-clean and hard-to-clean debris, specifically large liquid spillage debris. Larger debris is hard to clean for most of the floor-cleaning robots, thereby avoiding them is a better option rather than to spread them on the floor following a failed attempt to clean [1]. Recognizing and avoiding such larger debris is important for efficient cleaning by floor-cleaning robots. Typically, current floor-cleaning robots use piezoelectric debris sensors or optical sensors for recognizing debris on the floor. However, these sensing devices can identify debris only when it appears on the sensor and cannot be used for classifying debris types such as solid and liquid [1–3]. Hence, cleaning devices cannot avoid hard-to-clean liquid debris regions (e.g., large liquid spillage debris) and cannot predetermine the amount of cleaning effort required to achieve

better cleaning for small-sized liquid spillage debris [1]. Machine vision-based debris recognition is an emerging technique for autonomous cleaning robots. It provides an efficient solution for recognizing the debris on the floor [4–6]. Andersen et al. proposed a computer vision-based dusty area detection. The authors use a multivariable statistical method for recognizing the dirty areas and generate a cleanness map for the floor-cleaning robot [4]. Borman et al. proposed a debris-recognition technique for floor-cleaning robots where spectral residual image filter is used for detecting the dust and dirt debris on the floor [5]. Canny edge detection and Maximally Stable Extremal Regions (MSER)-based machine vision is proposed for floor-cleaning robots to recognize and classify the dust and mud detection on the floor [7]. Andreas et al. suggested an unsupervised learning-based dust and dirt detection algorithm. The authors use a Gaussian Mixture Model (GMM) to recognize the dust spots on the floor [8]. David et al. [9] developed a computer vision-based dust detection algorithm using median filter and thresholding technique where the median filter is used to remove the background of the surface and thresholding technique is used to recognize the debris on the floor. However, these techniques have shortcomings. The schemes can identify debris on the floor but cannot classify the debris types.

Deep-learning-based computer vision is an emerging technique for automatic recognition and classification. Deep-learning architectures have many hidden layers of neurons and the neural network architecture can be modified and optimized to solve different complex tasks in computer vision-based applications [10–17]. Chen et al. [10] proposed a computer vision-based robot grasping system for automatically sorting garbage where a Fast RCNN is employed for detecting different objects in the scene. Deep-learning-based head-pose detection is proposed by Jaishankar et al. for wearable pet robots [11]. The authors use AlexNet CNN architecture for implementing the head-pose detection system. Sa et al. [12] proposed a CNN-based fruit-detection system for an autonomous agricultural robotic platform. Here, the Faster RCNN framework is used for detecting the various kinds of fruit in the field. Saeed and Mathew developed a vision system for autonomous sewer robots using five-layer CNN architecture. The developed CNN architecture is used to detect and characterize cracks in a pipeline [13]. Li et al. proposed a surface defect detection algorithm. Here, Single-Shot MultiBox Detector (SSD) MobileNet network is employed to detect the surface defect [14]. Customized CNN architecture for debris classification and specific object-detection applications has been reported in [18,19]. However, customized CNN layers can heavily affect the object-detection accuracy and would not yield any better results when two object classes have very similar results. This can be overcome by combining CNN with other models such as SVM, which can solve simple classification problems such as classification based on size. This also provides better accuracy over fixed thresholds, while being lightweight, which enables real-time implementation. Recently, it was seen that some works included different combinations of both the CNN and the SVM to resolve some of the complex issues and improve the object detection and classification accuracy. One study carried out face detection using CNN [20]. SVMs were integrated with this model using kernel combination. Another study applied this technique to scene recognition by using features from different layers of CNN in the SVM [21]. The CNN-SVM classifier has also been used for recognizing handwritten digits [22]. In [23], Ucar et al. developed a hybrid object recognition and detection scheme for an autonomous driving vehicle by combining CNN and SVM technique where multiple CNNs are employed for determining local robust features and SVM recognizes and detects all objects. Even though there exists much literature demonstrating the benefits of deep-learning technique for various robotic application, none of this work is targeted towards debris classification for indoor floor-cleaning robots using deep-learning techniques. In this work, we present a debris-detection and classification task for floor-cleaning robot applications using cascaded SSD MobileNet and SVM frameworks. By adjusting the network parameters and structure, the deep-learning network can recognize and classify the debris as solids or liquids. Furthermore, the SVM classifier [24–26] is used to classify the CNN detected liquid spillage as smaller or larger regions. The experimental results demonstrate that the proposed cascaded scheme achieves better debris detection and classification capabilities than Faster RCNN

Inception architecture and achieves 97.5% classification accuracy. The CNN-SVM cascaded approach takes approximately 71 milliseconds for debris detection and classification on the captured floor image. It implies that the proposed technique can be implemented in real time for floor-cleaning robots to recognize debris and avoid mess which is hard to clean.

The remainder of this article is organized as follows: related work is reported in Section 2 and a brief introduction of CNN and SVM is given in Section 3. Section 4 describes the debris-detection and classification methodology in detail. The experimental results are given in Section 5. Finally, conclusions and future work are provided in Section 6.

2. Related Work

This section describes other literature on debris detection or classification. Both deep-learning and other techniques such as SVM have been used for garbage detection, but primarily only for outdoor situations with solid garbage. Also, most of these techniques consider only solid garbage. Gaurav et al. [27] developed a smartphone app to detect and localize the outdoor garbage. The authors used a pre-trained AlexNet CNN model for detecting garbage from outdoor captured images where Bing image search is used to collect the debris data set. The model has a classification accuracy of 87% for this application. However, their approach does not provide any information regarding the type of garbage and uses segmentation to detect regions, which is not very accurate. Another approach involved using an SVM with SIFT features to segregate garbage into recycling categories is provided in [19]. This method uses cropped images of solid garbage only for classification but does not detect the location of the trash. The method has an accuracy of 94% for the given task. The overfeat-googlenet model was trained with debris present in outdoor environments by Rad et al. [28]. The authors used 18,672 images of various types of litters and wastes to train CNN for identifying the outdoor environments solid debris such as leaves, newspapers, food packages, cans etc. The scheme achieved a precision of 68.27% for debris detection in this application. CNN architectures have also been used for underwater debris detection using FLS images [18]. This model was trained using images of common marine debris. Recently, Fulton et al. [15] evaluated various deep-learning frameworks for implementing the marine debris detectors in autonomous underwater vehicles. The authors realize that CNN and SSD have higher accuracy when compared to YOLOv2 and Tiny YOLO frameworks. However, SSD requires higher processing times.

3. Preliminary

3.1. Convolutional Neural Networks (CNN)

CNNs are very common in the field of image classification. They achieve high accuracy in many classification tasks by automatically detecting the features in a dataset of images. CNNs are better compared to traditional methods, where the filters that enable feature extraction have to be specified before computation. A typical CNN consists of four layers: a Convolution layer, Pooling layer, ReLU layer and Fully Connected layer. Each layer and its function is described here briefly. The CNN architecture, in general, is shown in Figure 1. A convolution layer extracts features, a pooling layer combines these features and reduces dimension, ReLU layers remove negative values, and the fully connected layers are responsible for classification.

The object-detection CNN architecture used in this paper comprises of two parts—a feature extractor and a bounding box predictor.

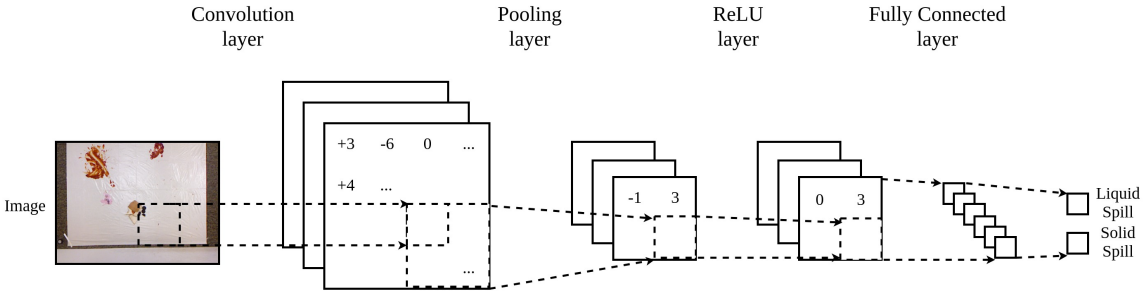


Figure 1. Structure of CNN.

3.2. Feature Extractors

Feature Extractors extract specialized features pertaining to a particular classification problem. These features are distinct for the given problem and can be used to distinguish between multiple classes. Specialized CNN architectures such as YOLO [29], MobileNet v2 [30], AlexNet [31], Inception v2 [32], and ResNet [33,34] are used for feature extraction [35]. MobileNet architecture is a prominent architecture for real-time scenarios. It uses depthwise separable convolution layers, which have a reduced computation cost compared to standard convolution layers. MobileNet v2 is an improved version of the MobileNet architecture. This architecture is highly efficient and can be deployed in low-power real-time scenarios. Inception is a popular architecture for feature extraction. It has a good balance between accuracy and cost but is far costlier than MobileNet. The architecture was developed after VGG and is more accurate and computationally efficient. It employs convolution layer factorization, which splits larger convolutions into a combination of smaller ones to achieve this. Inception v2 is used for a comparison of performance for the given task. ResNet is a state-of-the-art feature extractor, which obtains a high accuracy using residual learning. This improvement in accuracy is with 'shortcuts', which are the connections that skip levels. However, ResNets have a very high computation cost and cannot be employed in real-time systems.

3.3. Bounding Box Predictors

Bounding box predictors locate the objects in an image using feature maps extracted by the feature extractor. Two popular networks exist for this [35]—Faster RCNN [36] and SSD [37]. Faster RCNN is a commonly used algorithm for predicting bounding boxes. Faster RCNN is highly accurate. It uses a Region Proposal Network (RPN) to predict potential regions and a fully connected network for classification. This fully connected network uses both the feature map and the output of the RPN. Although Faster RCNN is efficient and accurate, it is not fast enough for real-time operations. SSD is more suited for real-time operations.

3.4. Support Vector Machine (SVM)

SVM is commonly used in classification problems [24–26]. SVM constructs a hyperplane as a separation between two or more classes. This hyperplane is constructed based on the distances from various points of different classes from it. The standard form of a hyperplane is given in Equation (1).

$$\vec{w}.\vec{x} - b = 0 \tag{1}$$

where $\vec{w}$ is a vector normal to the described hyperplane, referred to as the weight, and b represents the bias. For any data set containing points $(\vec{x}, y)$, with the class label y having two values $+1$ and -1, the hyperplane separating the two classes satisfies Equations (2) and (3).

$$\vec{w}.\vec{x_i} - b \geq 1 \quad \text{if} \quad y_i = +1 \tag{2}$$

$$\vec{w}.\vec{x_i} - b \leq 1 \quad \text{if} \quad y_i = -1 \tag{3}$$

However, this boundary may not be perfect. This can be due to the data not being completely separable. Hence, determining the hyperplane becomes a minimization problem where the hinge loss is minimized. Equation (4) describes the hinge loss function.

$$\sum max(0, 1 - y_i(\vec{w}.\vec{(x_i)} - b)) \tag{4}$$

Another scenario when the boundary may not be perfect is when the data is not linearly separable. Here, kernel functions such as Radial Bias Function (RBF), linear, Polynomial, Gaussian, etc., can be employed for classification with SVM.

4. Methodology

This section describes the enhanced deep-learning approach used for debris recognition and classification. Figure 2 illustrates the functional description of the proposed scheme.

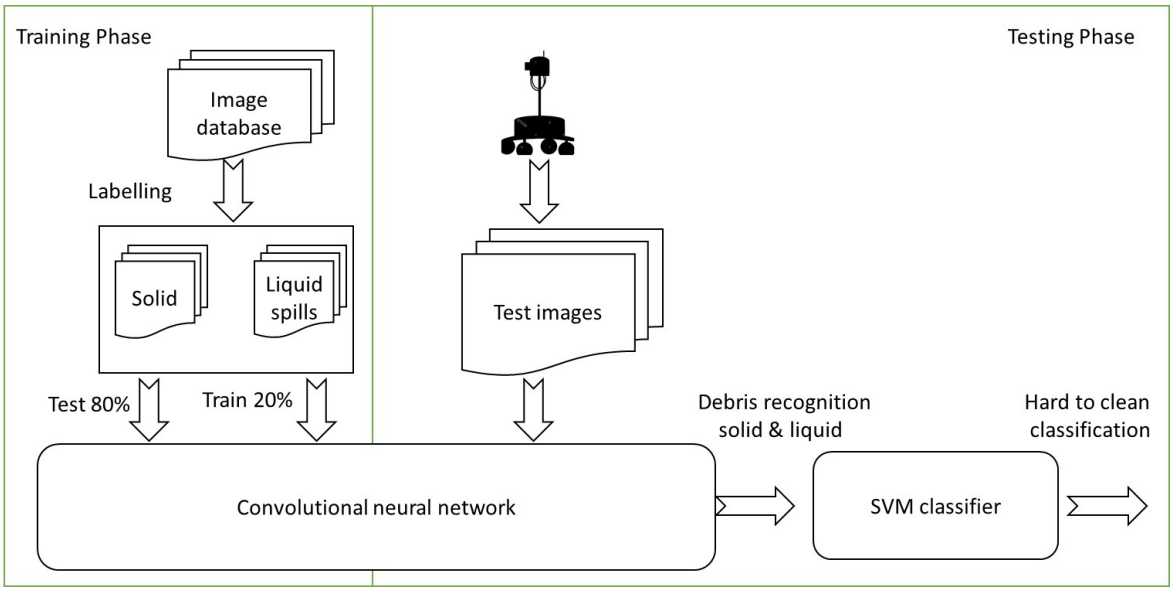


Figure 2. Proposed Scheme: Cascaded CNN and SVM technique.

The entire design flow is divided into two phases—the training phase and the testing phase. The proposed scheme uses MobileNet v2 for feature extraction and SSD for detection and classification of the floor debris into solid and liquid spillage debris. Then SVM is used to classify the liquid spillage debris regions based on spillage size as being small or large. Typically, identifying the size of the object from the boundary box outputs of CNN is the possible method, but this method faces several drawbacks. Firstly, it is based on the hard-coded threshold which is set manually and is hard to determine accurately [18,19]. Secondly, for finding the actual object size by boundary box method, we need very accurate intrinsic and extrinsic cameras parameters. The problem of camera calibration errors between the detected object in world frame and detected objects in the image frame are difficult to address in dynamic environments. Using SVM and with given training data, the input boundary box can be classified autonomously into two classes, i.e., hard-to-clean or easy-to-clean objects.

4.1. MobileNet V2 for Feature Extraction

MobileNet v2 is a lightweight feature extractor that can be run in real time on low-power embedded systems. A significant change over standard CNN architectures is the use of depthwise convolution layers. These layers have a lesser computation cost compared to standard convolution

layers. The depthwise separable convolution layers work on the principle of factorization. The standard convolution layers are factorized into two layers: a depthwise convolution layer and a pointwise convolution layer. This combination drastically reduces the computational complexities of the traditional convolution layers. Inverted residual layers are also used. This layer is based on the structure of convolution layers in the ResNet architecture i.e., residual learning which uses 'shortcuts'. These improve the accuracy of the depthwise convolution layer while not having a large overhead. Bottleneck layers that reduce the size of input are used as well. These reduce the computation time drastically. An upper bound is applied on the ReLU layer, which serves to limit the overall complexity. Structure of the standard convolution layers used in MobileNetv2 is shown in Figure 3.

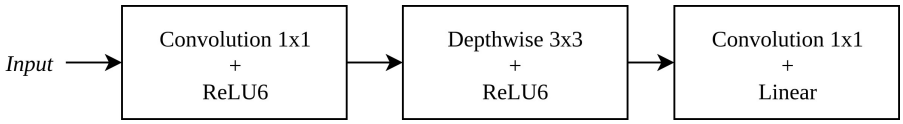


Figure 3. MobileNet v2 Architecture.

Since the location of debris is also required, a bounding box predictor is used in place of the fully connected layers present in the feature extractor.

SSD for Bounding Box Prediction

Bounding box predictors output the bounding box coordinates and type of class along with the confidence level of classification. These coordinates can be translated into real-world coordinates to help the floor-cleaning robot avoid larger spillage debris. SSD is used in this paper due to its low computation cost, thereby allowing it to be deployed in real-world systems such as small floor-cleaning robots. It classifies the debris into solids and liquids spillage.

SSD uses a CNN architecture to predict a set of bounding boxes over the region where a particular class is detected. These bounding boxes have a fixed size i.e., their sizes are selected from a set of predefined values. This modification helps to reduce the overall computation cost of these boxes. Although the boxes may not perfectly enclose the spill, the error is usually negligible. Since the boxes are of fixed sizes, the network employs multi-scale feature maps, which help to recognize objects of different scales. The network predicts a score along with the box coordinates, which indicates its confidence in the given prediction.

Since bounding boxes have both location and class type are parameters, the loss cannot be directly computed. A special measure must be used which is a combination of these parameters. The overall loss in the prediction of bounding boxes is given in Equation (5).

$$L = \frac{1}{N}(L_{conf} + \alpha L_{loc}) \tag{5}$$

where N is the number of boxes predicted and α is a weight parameter. L_{conf} is the SoftMax loss obtained from classification. L_{loc} is the L_1 error obtained from location estimation of each box.

4.2. Training Phase

Aver vision system and Kinect v2 RGB-D sensor are used in real-world scenarios to capture images for training the network. Two viewpoints were taken into consideration: top view and robot view. About 2000 training images are captured at dynamic floor backgrounds with various classes of food debris, including solid food waste, semi-solids, and large liquid spillage. Besides, to improve the CNN learning rate and prevent over-fitting, data expansion is applied to the captured images. Data Expansion increases the number of images in the dataset by using simple geometrical transformations such as rotation, scaling and flipping. Furthermore, to reduce the computing load in the training stage, the captured images are resized into 640×480 pixels. The dataset is balanced across the two classes. There are around 900 images for each class. These images contain one or more

debris of a single type (solid or liquid). The remaining images consist of mixed debris which contain both solid and liquid debris in the same image. The number of debris per image on an average is 2. These images are labeled manually for training the network by drawing bounding boxes over the detected debris.

The grid search algorithm determines some of the hyperparameters in the CNN and the hyperparameter λ for the SVM. It is using different combinations of possible values for finding the optimal hyperparameters. The combination that results in the highest accuracy is chosen. For CNN, the initial learning rate is set to 0.002. The decay factor for RMS prop is set to 0.9. A batch size of 32 is used and the number of epochs for training is 10,000. The other hyperparameters of the model are inherited from a pre-trained model (trained on MS COCO dataset) [35]. Similarly, the hyperparameter λ for SVM is obtained as 1.0.

A batch size of 20 is used while training the CNN. The loss is optimized through the Root Mean Squared Propagation (RMS Prop) [38] algorithm with a decay factor of 0.9. RMS prop uses gradient g_t, weight w_t^{rms} at any time t, exponential average v_t to estimate the weight at time $t+1$. Equation (8) is used to update the weights in the network. Here, η is the initial learning rate, set to 0.002. β is a hyperparameter tuned by the network. ϵ is a parameter introduced to prevent divide by zero errors.

$$v_t = \beta v_{t-1} + (1-\beta) g_t^2 \tag{6}$$

$$\Delta w^{rms} = \frac{-\eta}{\sqrt{v_t + \epsilon}} \times g_t \tag{7}$$

$$w_{t+1}^{rms} = w_t^{rms} + \Delta w^{rms} \tag{8}$$

The network is implemented in the TensorFlow framework on Ubuntu 16.04. A system with the following hardware configuration is used for training and testing: Intel Xeon E5-1600 V4 CPU with 64 GB RAM and an NVIDIA Quadro P4000 GPU with 12 GB Video memory.

The computation of performance metrics involves the use of K-fold cross-validation. In this technique, the dataset is divided into K subsets with K$-$1 subsets used for training and the remaining subset for evaluating the performance. This process is repeated K times to get the mean accuracy and other performance metrics of the model. K-fold cross-validation is done to ensure that the figures reported are accurate and not biased towards a particular dataset split. This work uses 10-fold cross-validation. The images shown are obtained from the model with the highest accuracy.

4.3. SVM for Error Reduction and Spill Size-Based Classification

Not all liquid spillage debris are hard to clean. While a robot may not be able to clean the solid debris, it should also not make a mess out of it. However, the same is not the case with liquid spills, where a classifier based on the spill size is required to prevent a mess and damage to cleaning equipment.

Here, SVM is employed to classify the size of liquid spillage debris detected. SVM would have a higher accuracy over a fixed threshold, which may not accurately determine the border cases. Also, SVM is lightweight on computational requirements, which helps for real-time implementation of this scheme. A soft margin SVM is used. The hyperplane boundary is modeled into a minimization problem shown in Equation (9).

$$f(x) = [\frac{1}{n} \times \sum max(0, 1 - y_i(\vec{m}.\vec{x_i} - b))] + \lambda ||\vec{m}||^2 \tag{9}$$

where $||\vec{m}||^2$ is a term used adjusting the margin size of a soft margin SVM classifier. Also, since the data is not linearly separable, a RBF kernel is used. The RBF kernel K on two points x_1 and x_2 is given in Equation (10).

$$K(x_1, x_2) = \exp\left(-\frac{||x_1 - x_2||^2}{2\sigma^2}\right) \tag{10}$$

where σ is a free parameter.

Two viewpoints are used in this paper for spill size detection, i.e., viewing from a top perspective and viewing from a robot's perspective. For both perspectives, three features—Area, Class, and Confidence are used in this classification.

Area: The area is used as a measure of the size of the spill. The area is estimated using a transformation as given in Equation (11).

$$Area = Width_{real} \times Height_{real} \tag{11}$$

$$Width_{real} = Width_{box} \times PPI \tag{12}$$

$$Height_{real} = Height_{box} \times PPI \tag{13}$$

$$PPI = \frac{Width_{image} \times Height_{image}}{Width_{area} \times Height_{area}} \tag{14}$$

where $Area$ is the estimated area, $Width_{real}$ and $Height_{real}$ are the real width and height computed from Equations (12) and (13). These equation use Pixel Per Inch(PPI), computed by Equation (14). $Width_{image}$ and $Height_{image}$ are the height and width of the captured image in pixels. $Width_{area}$ and $Height_{area}$ are the height and width of the area corresponding to the image in inches.

Class and Confidence: These measures are obtained from the SSD MobileNet network. These measures along with calculated area are used to filter out certain misclassifications or false detections such as floor patterns or small stains spread randomly on the floor. The Algorithms 1 and 2 represent the pseudocode for training and running the model of proposed scheme.

A pseudocode of the algorithm used for training the CNN and running the model is shown in Algorithms 1 and 2. The function $LOAD()$ loads the model into memory, $TRAIN()$ trains the model with the parameters specified. The remaining parameters are inherited from the model. The function $UPDATEMETRICS()$ updates the accuracy, precision, and recall based on the new predictions. The final accuracy, precision, recall are $acc, prec, rec$. The function $model_{SVM}$ is trained in a similar fashion but using only liquid spill size data.

Algorithm 1: Training the network pseudocode

Data: $input, labels, K$

$acc = 0; prec = 0; rec = 0; len = size(input)/K$

for $i = 0$ **to** K **do**

  $test_{input} = input[len \times i : len \times (i+1)]$

  $test_{labels} = labels[len \times i : len \times (i+1)]$

  $train_{input} = input - test_{input}$

  $train_{labels} = label - test_{labels}$

  $model = LOAD('ssd_mobilenetv2')$

  $model.TRAIN(train_{input}, train_{labels}, learningrate = 0.002, decay = 0.9, numepochs = 10000, batchsize = 32)$

  $predictions = model.PREDICT(test_{input})$

  $UPDATEMETRICS(acc, prec, rec, test_{labels}, predictions)$

end

$acc/ = K; prec/ = K; rec/ = K$

Algorithm 2: Simulation-optimization heuristic

Data: $image, model, model_{SVM}$
$predictions = model.PREDICT(image)$
for i ***in*** $RANGE(LEN(prediction))$ **do**
    $prediction.class = model_{SVM}.PREDICT(prediction.box.area, prediction.class, prediction.confidence)$
end

5. Experiments and Analysis

This section describes the experimental results of the proposed scheme. The proposed system should be able to detect and classify different types of debris. The scheme should also be able to classify liquid spillage debris based on their size. Hence, two experiments are conducted to validate this scheme. The first experiment is to assess the performance of debris classifier using real-world test images. The model has not been exposed to these images during the training phase. The second experiment is to assess the performance of liquid spill size classifier to avoid hard-to-clean debris.

5.1. Performance Metrics

Standard statistical measures are used to assess the performance of the proposed scheme. These include accuracy, precision, recall and $F_{measure}$ which are computed as shown in Equations (15)–(18). Here, tp, fp, tn, fn represents the true positives, false positives, true negatives, and false negatives respectively as per the standard confusion matrix. In Equations (19) and (20), η_{miss} is for the target debris not recognized by the network and η_{false} is for other objects detected as debris.

$$Accuracy(Acc) = \frac{tp + tn}{tp + fp + tn + fn} \tag{15}$$

$$Precision(Prec) = \frac{tp}{tp + fp} \tag{16}$$

$$Recall(Rec) = \frac{tp}{tp + fn} \tag{17}$$

$$F_{measure}(F_1) = \frac{2 \times precision \times recall}{precision + recall} \tag{18}$$

$$\eta_{miss} = \frac{\eta_{missnum}}{\eta_t estset} \times 100\% \tag{19}$$

$$\eta_{false} = \frac{\eta_{falsenum}}{\eta_t estset} \times 100\% \tag{20}$$

5.2. Debris Classification

Figures 4 and 5 shows the typical detection results for various types of floor debris captured at both top view and robot view with different angle. Here, solid debris is marked by a green rectangle box and liquid spills region are marked by a sky-blue rectangle box. The imaging system is mounted at the height of 150 cm from the floor for the top view and 60 cm for robot perspective .

The experimental results show that the trained SSD MobileNet architecture can detect most of the solid and liquid spill debris (97–98% of them) on the floor. Solid debris that is detected typically have a 98% or higher confidence level and liquid spillage debris have a 96% or higher confidence level, respectively. The overall accuracy of this classifier on both perspectives is shown in Table 1.

Table 1. AP and mAP for the proposed scheme.

Perspective	AP (%)		mAP (%)
	Solid Spills	Liquid Spills	
Top Perspective	99	97.5	98.25
Robot Perspective	98.5	94.2	96.4

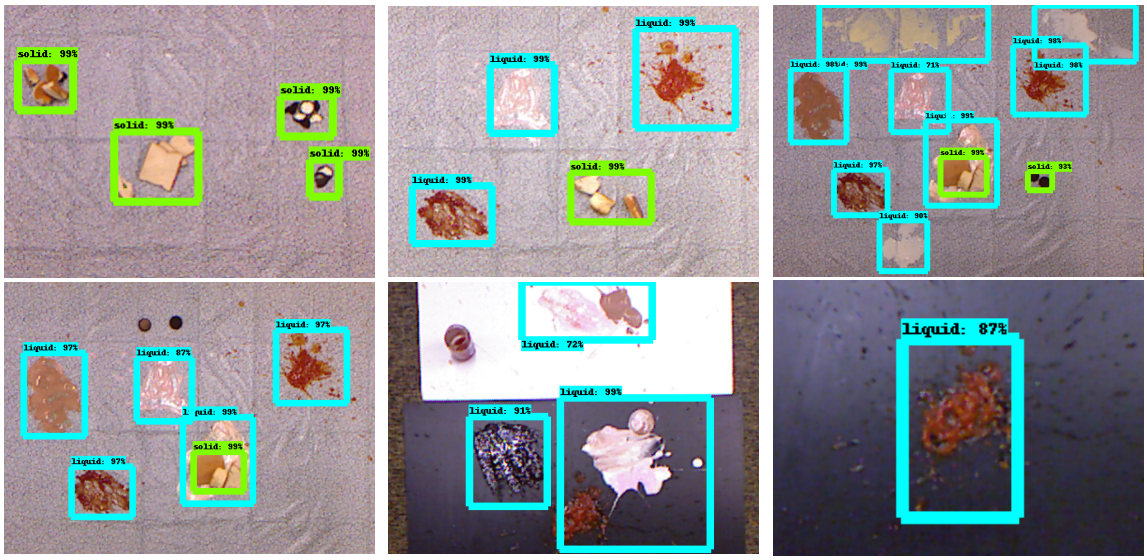


Figure 4. Overhead View Results.

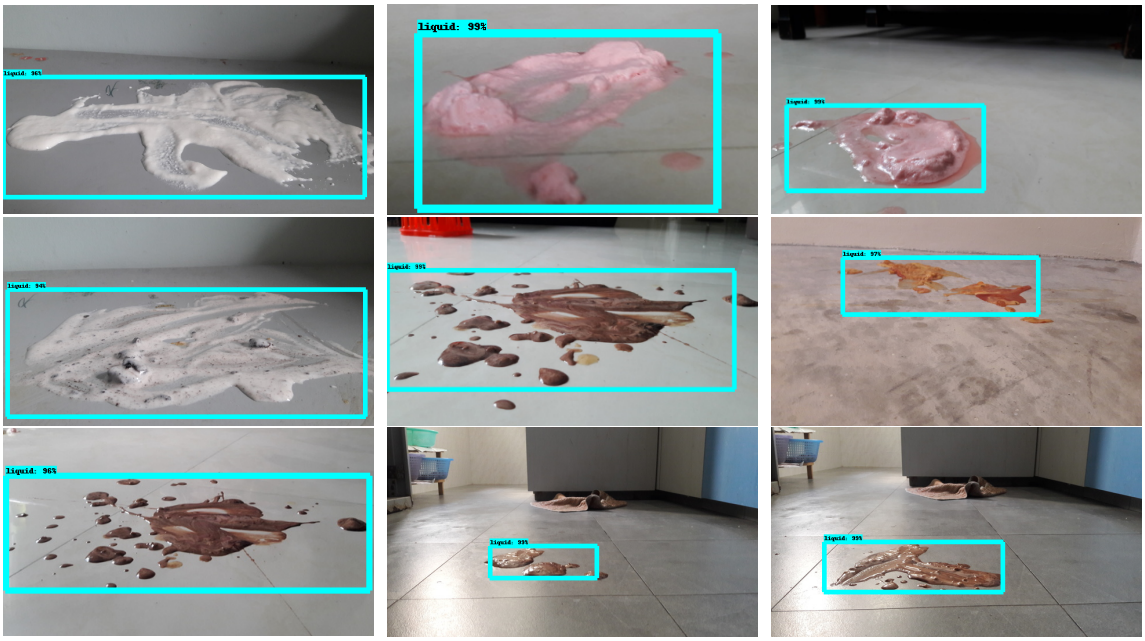


Figure 5. *Cont.*

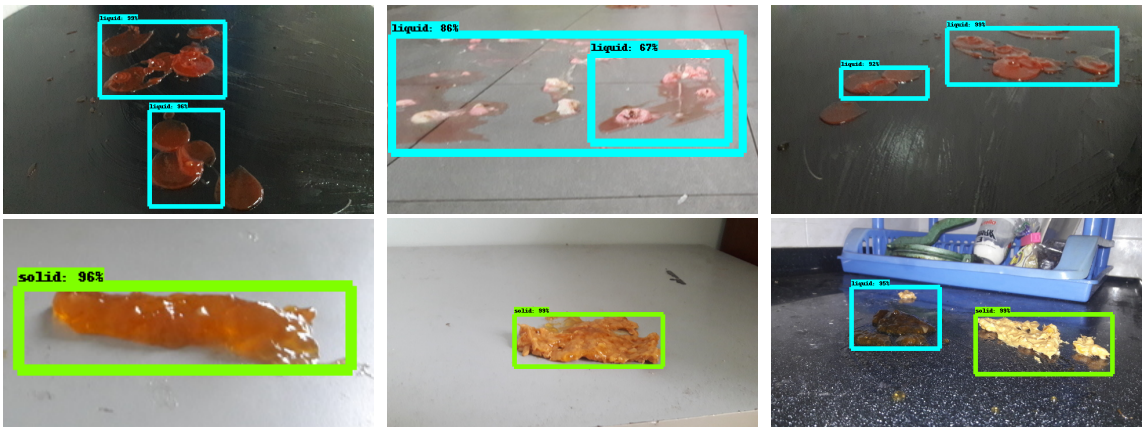


Figure 5. Front View Results.

Furthermore, to validate the robustness of the proposed scheme, debris classification has been applied to regions where a mixture of solid and liquid debris is present. The performance of the scheme on these scenarios is shown in Figure 6. The classification results indicate that the trained SSD MobileNet architecture can determine solid and liquid debris present in these scenarios very accurately.

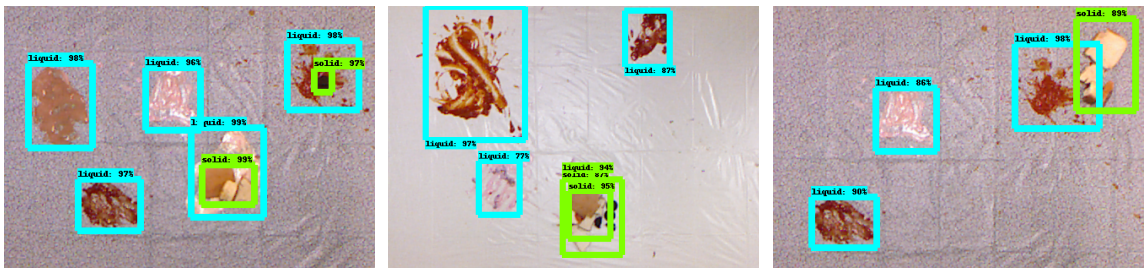
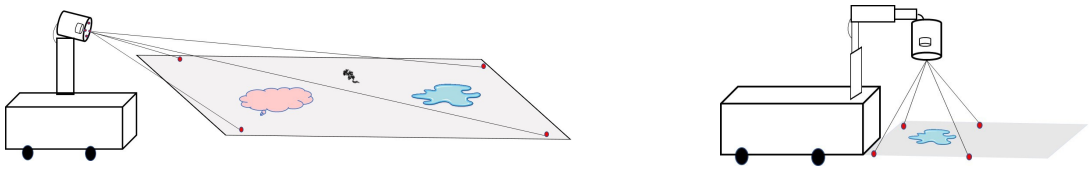


Figure 6. Detection of mixed debris.

5.3. Liquid Spill Size Classification

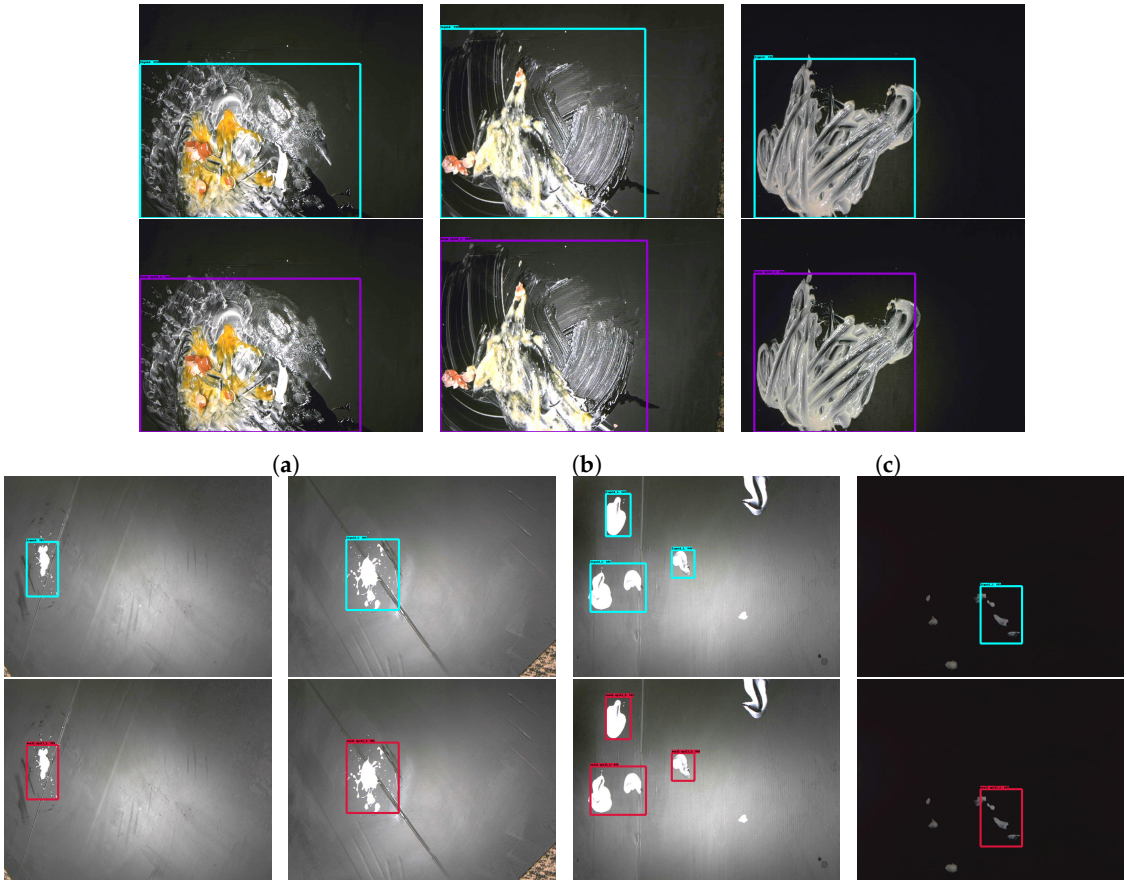
This experiment is performed with liquid spillage of varying compositions. Spillages made of everyday food products such as eggs, yogurt, milk, etc. are spread on the test bed with various sizes and captured by Aver Visualizer Module [39]. The Visualizer Module comprises of a 5 megapixel (MP), 180° Field of View CMOS camera sensor with a flexible arm for adjusting the camera height and tilt from −90–90° degrees as shown in Figure 7.

In our experiment, floor images are captured at 60 cm from the ground and the camera head is inclined at an angle of 45 degrees downwards for robot perspective (Figure 7a) and 90 degrees downwards (Figure 7b) for top perspective. The focused area has been measured accurately through a laser marker on-board the vision system. It helps to compute and verify the pixel-based bounding box area estimation. Then, the captured images are sent to CNN architecture for debris classification. Finally, the detected liquid spillage debris region is evaluated by an SVM to classify the liquid spillage debris based on size into large and small regions. Figure 8 shows the SVM classification results for various size of liquid spillage debris. A Violet bounding box indicates large liquid spillage debris and a red bounding box indicates small liquid spillage debris.



(**a**) (**b**)

Figure 7. Camera arrangement for capturing floor images in the experiments. (**a**) Robot Perspective; (**b**) Top Perspective.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**) (**g**)

Figure 8. Spill Size Classification Results. Turquoise box indicates CNN spill detection, Violet box indicates large spillage debris and red indicates small spillage debris. All areas are in cm^2. (**a**) Area: 1167.4; (**b**) Area: 1461.2; (**c**) Area: 888.8; (**d**) Area: 75.4; (**e**) Area: 158.4; (**f**) Area: 28.3, 111.3, 54.9; (**g**) Area: 89.1.

The experiment results show the efficacy of the proposed scheme for recognition and classification of commonly found debris in a home/food court environment. The SVM classifier achieved 91.6% classification accuracy. For large liquid spillage debris, the SVM classifier achieved 0.99 precision, 0.81 recall and an F-measure score of 0.89. For small liquid spillage debris, the model achieved 0.88 precision, 0.99 recall and an F-measure score of 0.94. The overall time taken by the SVM for an image on an average is 0.01 s.

For solid debris, the threshold is determined according to the size of vacuuming inlet of the cleaning device. The standard inlet size of commercial floor-cleaning robots is reported in Table 2. According to vacuuming inlet dimension, we can fix the threshold for solid objects to determine whether the detected objects should be avoided or cleaned. Furthermore, larger (in height) solid debris higher than ground clearance height (the gap between robot base and ground) of the robot is

considered to be an obstacle which can be detected by the bump sensor present in the robot and is automatically excluded from cleaning.

Table 2. Inlet Size of some common cleaning robots.

Manufacturer and Model	Cleaning Path Width (mm)
LG VR66820VMNC [40]	195
Samsung POWERbot VR 7000 [41]	280
iRobot-Roomba 675 [42]	177
Neato Robotics-Botvac D5 [43]	304
DEEBOT OZMO 930 [44]	359

5.4. Comparison of Performance of Different Architectures

To assess the efficiency of our proposed scheme, the classification accuracy of the proposed scheme has been compared with more robust networks such as Faster RCNN ResNet and Faster RCNN Inception CNN architectures. Both networks have been trained using the same dataset for a similar amount of time.

The metrics indicate that Faster RCNN ResNet performs the best compared to the other model. Faster RCNN Inception generally performs better compared to SSD MobileNet but not in the scenario of debris type classification. In contrast to the other schemes, SSD MobileNet uses a significantly lesser amount of time for each image. Also, SSD MobileNet is optimized for low powered hardware which makes it possible to deploy the network in real-time situations using low powered computing hardware such as Intel Movidius Neural Compute Stick and Raspberry Pi. Hence, the proposed scheme can be implemented on a small cleaning robot for real-time debris detection and classification.

The garbage detection and classification results for Faster RCNN Resnet, Faster RCNN Inception and the proposed scheme are shown in Table 3. The comparison is made using standard metrics discussed above. Also, timing analysis has been performed to estimate the average detection time of each network. It has been tested on NVIDIA Quadro P4000 graphic card and reported in Table 4. The comparison of predictions results of three schemes on two given images is shown in Figure 9.

Table 3. Performance of different models for debris detection and classification.

Model	Liquid			Solid			Overall		
	Prec	Rec	F_1	Prec	Rec	F_1	η_{miss}	η_{false}	Acc
Faster RCNN ResNet	96.9	99.4	98.1	98.6	93.0	95.7	3	1	97.8
Faster RCNN Inception	91.1	97.1	94.0	93.8	82.2	87.6	7	3	91.9
Proposed Scheme—SSD MobileNet	94.2	99.3	96.7	98.5	87.8	92.8	4	2	95.5

Table 4. Execution time comparison.

Model	Time Taken for 60 Images (s)	Average Time per Image (ms)
Faster RCNN ResNet	11.07	184
Faster RCNN Inception	8.29	138
Proposed Scheme—SSD MobileNet	4.28	71

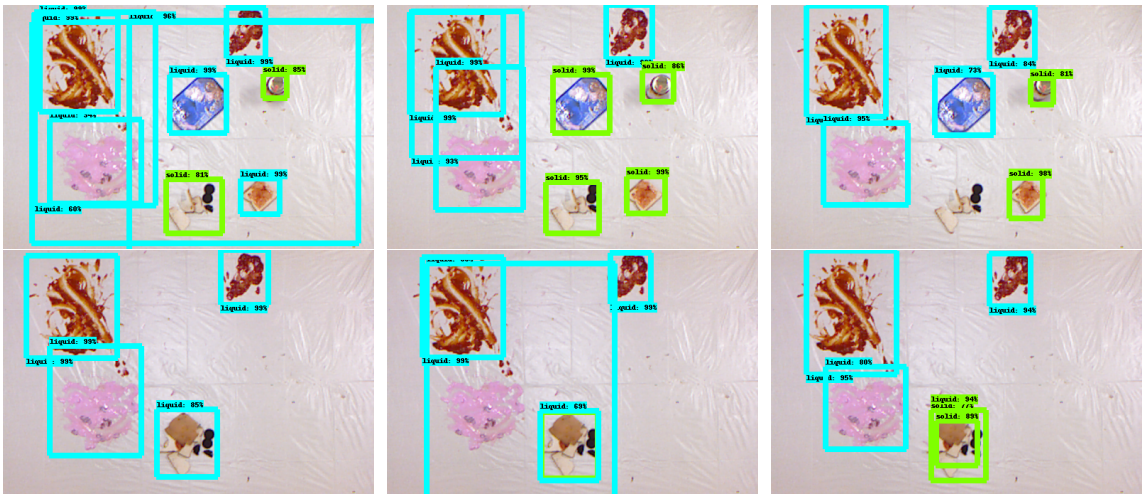


Figure 9. Predictions of three networks, Faster RCNN Resnet (**Left**), Faster RCNN Inception (**Center**), Proposed Scheme (**Right**) on the same images.

5.5. Comparison with Existing Schemes

This section describes the differences analysis of debris detection and novel-object-detection application using deep-learning technique. Table 5 shows the difference in proposed implementation with existing CNN-based debris-detection and classification schemes. The difference has been listed based on CNN architecture, network hyperparameter, and object-detection accuracy. Faster RCNN architecture has higher accuracy compared to SSD for debris detection and other object-detection scenarios for different feature extractors. However, SSD is faster because it uses a defined set of sizes for the bounding box and uses the more-efficient depthwise convolution layers. These contribute to a lower computation cost but results in a reduction of accuracy. YOLO-based networks are faster than SSD but have a poor accuracy in most tasks. A comparison of the performance of these networks is shown in Table 6. SSD MobileNet provides a good balance between accuracy and computation cost and is fast enough to be used in real-time frameworks for object detection and specifically debris classification. Comparison with non-deep-learning-based approaches is shown in Table 7. Deep-learning techniques offer a large advantage in scenarios when sufficient data is available. These techniques can autonomously extract features from images, which allow them to learn features and patterns which are difficult to figure out statistically. CNN architectures are specifically good for extracting features from images since they use a combination of pixels next to each other for features.

Table 5. Comparison with Similar Schemes.

Scheme	Framework	Training Dataset Size	Batch Size	Decay	Initial Learning Rate	mAP (%)	Execution Time (s)
Gaurav et al. [27]	AlexNet	2451	100	0.0005	0.01	87.69	1.50
Yang et al. [19]	CNN (11 layer)	2400	32	0.5	0.5	22.0	-
Rad et al. [28]	OverFeat GoogleNet	18,676	16	-	-	63.2	-
Chen et al. [10]	Fast RCNN	1999	32	0.0005	-	-	0.22
Proposed	MobileNet SSD	2000	32	0.9	0.002	96.4	0.071

Table 6. Comparison with DL Schemes for Other Applications.

Scheme	Framework	Training Dataset Size	Epochs	mAP (%)	Miss Rate (%)	False Rate (%)	Execution Time (s)	Application
Li et al. [14]	MobileNet SSD	400	10,000	95	-	2	0.12	Material Surface Defect Detection
Saeed et al. [13]	CNN (3 layer)	160 video frames	1000	-	-	-	-	Pipe Joint Detection
Fulton et al. [9]	YOLO v2	5720	-	47.9	-	-	0.11	Marine Debris Detection
	tiny YOLO		-	31.6	-	-	0.52	
	Faster RCNN Inception v2		-	81.0	-	-	0.97	
	SSD MobileNet		-	67.4	-	-	3.19	
Proposed	MobileNet SSD	2000	10,000	96.4	4	2	0.071	Debris Detection and Classification

Table 7. Comparison with schemes not based on deep learning.

Scheme	Algorithm	Accuracy
Bormann et al. [5]	Spectral residual filter	75.45
Milinda et al. [7]	Spectral residual filter + Maximally Stable Extremal Regions	80.12
Mittal et al. [27]	HOG + Gabor + Color	80.32
Yang et al. [19]	SVM	63
Proposed	MobileNet SSD	95.5

6. Conclusions and Future Work

This work proposed debris detection and liquid spillage debris classification based on size for floor-cleaning robot application using cascaded machine-learning algorithms involving SSD Mobile net deep-learning framework and SVM classifier. The feasibility of the proposed method was verified with real debris images captured with the top perspective and the robot perspective. The experimental results show that the proposed scheme can identify and classify both solid and liquid spillage debris with more than 96% accuracy in both perspectives and achieves better detection and classification performance than existing deep-learning schemes and other computer vision-based debris classification techniques. Furthermore, for deploying in the real floor-cleaning robotic platform, the robot perspective has added advantages than top perspective. Therefore, in most of the vision system-enabled floor-cleaning devices, the vision modules are fixed in robot perspective and can cover the large floor area (approximately 400 cm coverage area from current position). These factors make the robot perspective a better approach for spill detection, which can be easily deployed in any specific existing or new platform.

As the proposed technique has the potential to be implemented in real-world applications, we plan to deploy this scheme in reconfigurable robots being developed at SUTD, Singapore [45,46]. Furthermore, we plan to bridge the vSLAM algorithm with the proposed scheme to resolve the occlusion issues in real-time implementation on floor-cleaning robots. Through vSLAM techniques, robots can compute the debris location and update the debris location in vision-based mapping systems, which helps track the detected object even if it is occluded by another object.

Author Contributions: B.R., A.K.L., M.I. and M.R.E. designed the algorithm, and carried out the experiment, analyzed the result and wrote the paper; M.R.E. and A.V.L. analyzed the data and gave helpful suggestion on this research.

Funding: This work is financially supported by the National Robotics R&D Program Office, Singapore, under the Grant No. RGAST1702.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Park, J.H.; Park, D.R. Dust Detection Method and Apparatus for Cleaning Robot. U.S. Patent 7,920,941, 5 April 2011.
2. Hess, J.; Beinhofer, M.; Kuhner, D.; Ruchti, P.; Burgard, W. Poisson-driven dirt maps for efficient robot cleaning. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 6–10 May 2013; pp. 2245–2250.
3. Lee, H.; Banerjee, A. Intelligent scheduling and motion control for household vacuum cleaning robot system using simulation based optimization. In Proceedings of the Winter Simulation Conference (WSC), Huntington Beach, CA, USA, 6–9 December 2015; pp. 1163–1171.
4. Andersen, N.A.; Braithwaite, I.D.; Blanke, M.; Sorensen, T. Combining a novel computer vision sensor with a cleaning robot to achieve autonomous pig house cleaning. In Proceedings of the 44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference, Seville, Spain, 15 December 2005; pp. 8331–8336.

5. Bormann, R.; Fischer, J.; Arbeiter, G.; Weisshardt, F.; Verl, A. A visual dirt detection system for mobile service robots. In Proceedings of the ROBOTIK 2012; 7th German Conference on Robotics, Munich, Germany, 21–22 May 2012; pp. 1–6.
6. Bormann, R.; Weisshardt, F.; Arbeiter, G.; Fischer, J. Autonomous dirt detection for cleaning in office environments. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 6–10 May 2013; pp. 1260–1267.
7. Milinda, H.; Madhusanka, B. Mud and dirt separation method for floor cleaning robot. In Proceedings of the 2017 Moratuwa Engineering Research Conference (MERCon), Moratuwa, Sri Lanka, 29–31 May 2017; pp. 316–320.
8. Grünauer, A.; Halmetschlager-Funek, G.; Prankl, J.; Vincze, M. The Power of GMMs: Unsupervised Dirt Spot Detection for Industrial Floor Cleaning Robots. In *Conference Towards Autonomous Robotic Systems*; Springer: Berlin, Germany, 2017; pp. 436–449.
9. Martínez, D.; Alenya, G.; Torras, C. Planning robot manipulation to clean planar surfaces. *Eng. Appl. Artif. Intell.* **2015**, *39*, 23–32. [CrossRef]
10. Zhihong, C.; Hebin, Z.; Yanbo, W.; Binyan, L.; Yu, L. A vision-based robotic grasping system using deep learning for garbage sorting. In Proceedings of the 2017 36th Chinese Control Conference (CCC), Dalian, China, 26–28 July 2017; pp. 11223–11226.
11. Bharatharaj, J.; Huang, L.; Mohan, R.; Pathmakumar, T.; Krägeloh, C.; Al-Jumaily, A. Head Pose Detection for a Wearable Parrot-Inspired Robot Based on Deep Learning. *Appl. Sci.* **2018**, *8*, 1081. [CrossRef]
12. Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. Deepfruits: A fruit detection system using deep neural networks. *Sensors* **2016**, *16*, 1222. [CrossRef] [PubMed]
13. Shiry, S.; Center, C.; Browne, M.C. Convolutional Neural Networks for Robot Vision: Numerical Studies and Implementation on a Sewer Robot. Available online: http://staff.itee.uq.edu.au/lovell/aprs/anziis2003/Papers/paper133.pdf (accessed on 31 October 2018).
14. Li, Y.; Huang, H.; Xie, Q.; Yao, L.; Chen, Q. Research on a Surface Defect Detection Algorithm Based on MobileNet-SSD. *Appl. Sci.* **2018**, *8*, 1678. [CrossRef]
15. Fulton, M.; Hong, J.; Islam, M.J.; Sattar, J. Robotic Detection of Marine Litter Using Deep Visual Detection Models. *arXiv* **2018**, arXiv:1804.01079.
16. Tang, C.; Ling, Y.; Yang, X.; Jin, W.; Zheng, C. Multi-View Object Detection Based on Deep Learning. *Appl. Sci.* **2018**, *8*, 1423. [CrossRef]
17. Kim, K.; Hong, S.; Choi, B.; Kim, E. Probabilistic Ship Detection and Classification Using Deep Learning. *Appl. Sci.* **2018**, *8*, 936. [CrossRef]
18. Valdenegro-Toro, M. Submerged marine debris detection with autonomous underwater vehicles. In Proceedings of the 2016 International Conference on Robotics and Automation for Humanitarian Applications (RAHA), Kollam, India, 18–20 Deccember 2016; pp. 1–7.
19. Yang, G.T.M.; Thung, G. Classification of Trash for Recyclability Status. CS229 Project Report; 2016. Available online: https://pdfs.semanticscholar.org/c908/11082924011c73fea6252f42b01af9076f28.pdf (accessed on 31 October 2018).
20. Tao, Q.-Q.; Zhan, S.; Li, X.-H.; Kurihara, T. Robust face detection using local CNN and SVM based on kernel combination. *Neurocomputing* **2016**, *211*, 98–105. [CrossRef]
21. Tang, P.; Wang, H.; Kwong, S. G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition. *Neurocomputing* **2017**, *225*, 188–197. [CrossRef]
22. Niu, X.X.; Suen, C.Y. A novel hybrid CNN-SVM classifier for recognizing handwritten digits. *Pattern Recognit.* **2012**, *45*, 1318–1325. [CrossRef]
23. Uçar, A.; Demir, Y.; Güzeliş, C. Object recognition and detection with deep learning for autonomous driving applications. *Simulation* **2017**, *93*, 759–769. [CrossRef]
24. Wang, W.-C.; Chen, L.-B.; Chang, W.-J. Development and Experimental Evaluation of Machine-Learning Techniques for an Intelligent Hairy Scalp Detection System. *Appl. Sci.* **2018**, *8*, 853. [CrossRef]
25. Elangovan, K.; Krishnasamy Tamilselvam, Y.; Mohan, R.; Iwase, M.; Takuma, N.; Wood, K. Fault Diagnosis of a Reconfigurable Crawling—Rolling Robot Based on Support Vector Machines. *Appl. Sci.* **2017**, *7*, 1025. [CrossRef]

26. Meirista, E.; Mukhlash, I.; Setiyono, B. Watermelon Plant Classification Based on Shape and Texture Feature Leaf Using Support Vector Machine (Svm). In Proceedings of the International Conference on Research, Implementation and Education of Mathematics and Sciences 2015 (ICRIEMS 2015), Yogyakarta State University, Special Region of Yogyakarta , Indonesia, 17–19 May 2015; ISSN 978-979-96880-8-8.
27. Mittal, G.; Yagnik, K.B.; Garg, M.; Krishnan, N.C. Spotgarbage: Smartphone app to detect garbage using deep learning. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, Heidelberg, Germany, 12–16 September 2016; pp. 940–945.
28. Rad, M.S.; von Kaenel, A.; Droux, A.; Tieche, F.; Ouerhani, N.; Ekenel, H.K.; Thiran, J.-P. A Computer Vision System to Localize and Classify Wastes on the Streets. In *International Conference on Computer Vision Systems*; Springer: Cham, Switzerland, 2017; pp. 195–204.
29. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
30. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. *arXiv* **2018**, arXiv:1801.04381.
31. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
32. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016; pp. 2818–2826.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016; pp. 770–778.
34. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. *AAAI* **2017**, *4*, 12.
35. Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S.; et al. Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
36. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA, 3–6 December 2012; pp. 91–99.
37. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Berlin, Germany, 2016; pp. 21–37.
38. Tieleman, T.; Hinton, G. *Lecture 6.5-RMSProp, COURSERA: Neural Networks for Machine Learning*; Technical Report; University of Toronto: Toronto, ON, USA, 2012
39. Aver Visualizer Module. Available online: http://presentation.aver.com/model/355AF (accessed on 31 October 2018).
40. LG VR66820VMNC. Available online: https://www.bestdenki.com.sg/robotic-vacuum/lg/vr66820vmnc (accessed on 31 October 2018).
41. Samsung POWERbot VR 7000. Available online: https://stacksocial.com/sales/samsung-powerbot-star-wars-limited-edition-stormtrooper-robot-vacuum (accessed on 31 October 2018).
42. iRobot—Roomba 675. Available online: https://www.google.com/url?q=https://www.bestbuy.com/site/irobot-roomba-675-app-controlled-self-charging-robot-vacuum-black/6280532.p%3FskuId%3D6280532&ust=1543841220000000&usg=AFQjCNES1JzbsCqD_e8ao-bcNm_-niTyPw&hl=en&source=gmail (accessed on 31 October 2018).
43. Neato Robotics—Botvac D5. Available online: https://www.bestbuy.com/site/neato-robotics-botvac-d5-app-controlled-robot-vacuum-black/5559900.p?skuId=5559900 (accessed on 31 October 2018).
44. DEEBOT OZMO 930. Available online: https://www.bestbuy.com/site/ecovacs-robotics-deebot-ozmo-930-app-controlled-self-charging-robot-vacuum-mop-black/6215304.p?skuId=6215304 (accessed on 31 October 2018).

45. Prabakaran, V.; Elara, M.R.; Pathmakumar, T.; Nansai, S. hTetro: A tetris inspired shape shifting floor cleaning robot. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 6105–6112.
46. Ilyas, M.; Yuyao, S.; Mohan, R.E.; Devarassu, M.; Kalimuthu, M. Design of sTetro: A Modular, Reconfigurable, and Autonomous Staircase Cleaning Robot. *J. Sens.* **2018**, *2018*. [CrossRef]