A Study of Garbage Classification with Convolutional Neural Networks

Shanshan Meng
Humboldt University of Berlin
Berlin, Germany
meng_shanshan@outlook.com

Wei-Ta Chu
National Cheng Kung University
Tainan, Taiwan
wtchu@gs.ncku.edu.tw

***Abstract*—Recycling is already a significant work for all countries. Among the work needed for recycling, garbage classification is the most fundamental step to enable cost-efficient recycling. In this paper, we attempt to identify single garbage object in images and classify it into one of the recycling categories. We study several approaches and provide comprehensive evaluation. The models we used include support vector machines (SVM) with HOG features, simple convolutional neural network (CNN), and CNN with residual blocks. According to the evaluation results, we conclude that simple CNN networks with or without residual blocks show promising performances. Thanks to deep learning techniques, the garbage classification problem for the target database can be effectively solved.**

I. Introduction

Currently, the world generates 2.01 billion tons of municipal solid waste annually, which is huge damage to the ecological environment. Waste production will increase by 70% if current conditions persist [1]. Recycling is becoming an indispensable part of a sustainable society. However, the whole procedure of recycling demands a huge hidden cost, which is caused by selection, classification, and processing of the recycled materials. Even though consumers are willing to do their own garbage sorting nowadays in many countries, they might be confused about how to determine the correct category of the garbage when disposing of a large variety of materials. Finding an automatic way to do the recycling is now of great value to an industrial and information-based society, which has not only environmental effects but also beneficial economic effects.

Since 2006 the industry of artificial intelligence has welcomed its third wave with sufficient database. Deep learning began to show its high efficiency and low complexity in the field of computer vision. Many new ideas were proposed to gain accuracy in image classification and object detection. Among various deep models, convolutional neural networks (CNNs) [2] [3] especially have led to a series of breakthroughs for image classification. CNNs capture features of images with "strong and mostly correct assumptions about the nature of images" [2]. Owing to the fewer connections of CNNs in comparison to fully connected neural networks, CNNs are easier to be trained with fewer parameters. Therefore, in this paper, we would like to investigate different models based on convolutional neural networks to do garbage classification. Overall, this study is to identify a single object in an image and to classify it into one of the recycling categories, such as mental, paper, and plastic.

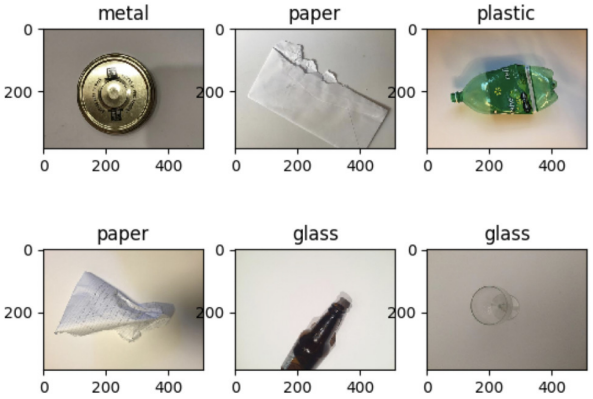

Fig. 1. Sample images of the garbage classification dataset.

The rest of this paper is organized as follows. Sec. II describes the garbage image dataset. Details of studied models are described in Sec. III. Sec. IV provides comprehensive evaluation studies and discussion, followed by conclusion of this work in Sec. V.

II. Dataset

For garbage classification, we utilize the images of the dataset dedicated to the garbage classification task on Kaggle[1]. This dataset includes totally 2527 images in which a single object of garbage is present on a clean background. Lighting and pose configurations for objects in different images is different. All these images have the size of 384×512 pixels and belong to one of the six recycling categories: cardboard, glass, metal, paper, plastic, and trash.

To train deep neural networks, we need a large amount of training images. With flipping and rotation, we augment the dataset to 10108 images, which was randomly split into train sets of 9,095 images and test sets of 1,013 images. Some sample images in this dataset are shown in Fig. 1.

978-1-7281-4999-8/20/$31.00 ©2020 IEEE

[1]https://www.kaggle.com/asdasdasasdas/garbage-classification

III. Methodology

A. HOG + Support Vector Machine

Since all the objects were placed on a clean background, we firstly try to capture gradient features of images and then construct a classifier based on support vector machine (SVM) to do classification.

The gradient features we employ are histogram of oriented gradients (HOG) [4]. The distribution of gradients of different directions can somehow describe appearance and shape of objects within an image. The HOG descriptor is invariant to geometric and photometric transformations. The image is divided into small rectangular regions and the HOG features are compiled in each region. The oriented gradients of each cell are counted in 9 histogram channels. After the block normalization using L2-Norm with limited maximum values, the feature vectors of cell histograms are concatenated to a feature vector of the image.

The extracted feature vectors are fed to an SVM, which is a canonical classification method before the era of deep learning. An SVM classifier is constructed by finding a set of hyperplanes between different classes in a high-dimensional space. The learning algorithm attempts to find the hyperplane that has the largest total distance to the nearest training data point of any class, which means the lowest error of the classifier at the same time.

B. Simple CNN Architecture

To investigate performance of a basic CNN, we build a simple CNN architecture to get general inspection, which may help to realize the performance difference between models. This architecture uses 2D convolutional (conv. in short) layers to capture features of images. Since filters of size 3×3 allow more applications of nonlinear activation functions and decrease the number of parameters than larger filters [5], the built simple CNN model uses 3×3 filters for all the conv. layers. Between 2D conv. layers we add the max pooling layers to reduce dimensions of the input and the number of parameters to be learned. This could preserve important features after conv. layers while preventing overfitting. After the conv. blocks there is a flatten layer, which flattens the feature matrix into a column vector. This allows the model to use two fully-connected layers at the end to do the classification.

In this architecture, we use two activation functions. In all the conv. layers and after the flatten layer we use the Rectified Linear Unit function (ReLU) defined as $y = \max(0, x)$ to introduce nonlinearity into the model, which could avoid the problem of gradient vanishing during back-propagation and has a lower calculation complexity. In the last dense layer, we use the softmax function as activation, which fits the cross-entropy loss function well. Fig. 2 illustrates structure of the simple CNN.

C. ResNet50

In empirical experiments [6], researchers found that very deep convolutional neural networks are difficult to train. The accuracy may become overly saturated and suddenly degrade.

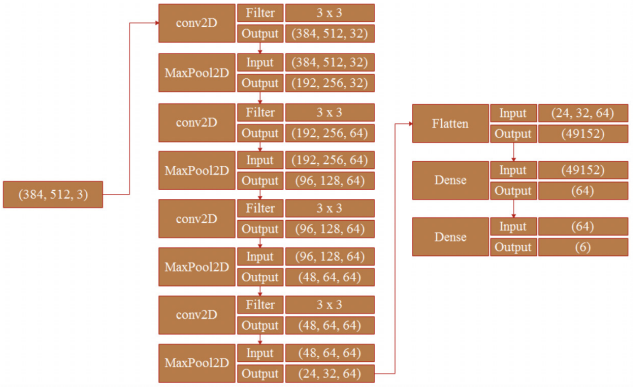

Fig. 2. Structure of the simple CNN.

Therefore, the residual network was proposed to diminish this problem.

In the ResNet proposed in [6], the residual block tries to learn the residual part of the true output. It uses the shortcut connection of identity mapping to add earlier parts of the network into the output. Such shortcuts won't add extra parameters or extra complexity. But the residual part is much easier to be trained than original functions in empirical experiments. In a variant of the ResNet, called ResNet50, researchers use the bottleneck architecture in the residual block. In each residual block, there are two conv. layers with a filter of size 1×1 before and after the normal 3×3 conv. layer. These 1×1 conv. layers reduce and then increase dimensions, which "leave the 3×3 layer a bottleneck with smaller input/output dimensions" [6] and keep the same dimensions of the identity part and the residual part.

In the model of ResNet50, we firstly use a conv. layer and a pooling layer to get the rough features of images. After the normal conv. block, the model uses totally 16 residual blocks with an increasing dimension of features. The last residual block is connected with an average pooling layer to downsample the feature matrix, a flatten layer to convert the feature matrix into a vector, a dropout layer and a fully connected layer to classify the features of an image into one category. The dropout layer, considered to be a way of regularization, can not only add noise to the hidden units of a model, but can also average the overfitting errors and reduce the co-adaptions between neurons.

The residual blocks also use ReLU as activation function to make the most of its advantages. The same as the simple CNN architecture, ResNet50 also uses softmax as the activation function in the last layer. Fig. 3 illustrates structure of the ResNet50 model.

D. Plain Network of ResNet50

To make a comparison between models with and without residual blocks, we also build a plain network of ResNet50 without the identity shortcuts. This plain network still contains the bottleneck block, which acts on the changing of dimensions and reduction of parameters. Without the identity

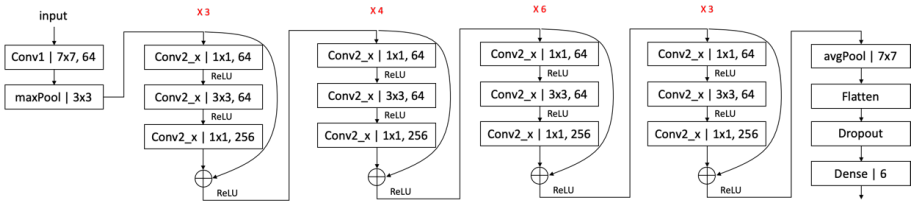

Fig. 3. Structure of the ResNet50 model.

mapping, this model is constructed based on the original function instead of the residual function. The size of the filter, dimensions of feature matrix and selection of activation functions of plain network are the same with ResNet50. Fig. 4 illustrates structure of the plain ResNet50 model.

E. HOG+CNN

We are also wondering the performance if we combine traditional hand-crafted features with CNN features. Therefore, we build a new network to jointly consider two types of features. This network has two parts at the first stage: the convolutional part and HOG part. The convolutional part includes 4 conv. layers with max pooling layers (similar to structure of the simple CNN model). The HOG part firstly resizes the image into 200×200 pixels. It then extracts HOG features of the image with L2-Normalization. Concatenation of flattened CNN features and HOG features is fed to three fully connected layers. These dense layers are followed by a dropout and another dense layer to do the classification. This model uses ReLU as the activation function for all connections except that the last dense layer uses the softmax activation. Fig. 4 illustrates structure of the hybrid model.

F. Loss Function and Optimizer

For all the four CNN models mentioned above, we use the cross entropy as the loss function. The cross-entropy loss function measures the subtle differences between classification results. Based on the loss function, we can find the optimal parameter settings by the gradient descent algorithm.

For the aforementioned CNNs, we use both the Adam optimizer and the Adadelta optimizer to see the differences. The Adam optimizer is seen as a combination of RMSprop and momentum. It computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. This has the effect of making the algorithm more efficiently reach convergence given lots of data. The Adadelta optimizer is a general situation of RMSprop. It restricts the window of accumulated past gradients to some fixed size instead of summing up all past squared gradients (like Adagrad), which avoids early stop of learning caused by gradient vanishing.

IV. Evaluation

A. Experimental Settings

To construct the SVM classifier, the radial basis kernel is used for feature projection, and the libSVM library [7] is used for implementation.

TABLE I
Performance of the simple CNN architecture.

Optimizers	Training Accuracy	Test Accuracy
Adam	92.55%	90.69%
Adadelta	94.74%	93.75%

The experiment with the ResNet50 model employs the pre-trained weights of the model that was trained on ImageNet dataset. For the simple CNN and HOG+CNN models, the weights were randomly initialized. For ResNet50, plain network of ResNet50, and the HOG+CNN models the ratio of dropout layer is all set at 0.5.

To get a more accurate description of the models, the dataset is split randomly for 3 times. All the models are trained with the shuffled dataset of 9,095 train images and 1,013 test/validation images for 40 epochs. The results showed below are the average of all the experiments. Due to our hardware limitation, the simple CNN architecture is trained with a batch size of 32, and ResNet50, plain network, and HOG+CNN models are with 16.

B. Experimental Results

1) Support Vector Machine: The SVM-based approach achieves test accuracy around 47.25% using the same training and test sets with other models. The HOG features may not describe the features very precisely. Only moderate classification performance can be obtained, given that only six categories are to be classified. Therefore, this method can be taken as a baseline for further comparison.

2) Simple CNN Architecture: Table I shows classification performance of the simple CNN architecture. Results obtained based on two optimizers are compared. As can be seen, using the Adadelta optimizer yields slightly better training and test accuracies.

Fig. 6 shows the evolutions of training/test accuracies and training/test losses as the number of epochs increases. The simple model achieves a training accuracy over 94% and test accuracy over 93% using a 90/10 training/testing data split with the Adadelta optimizer. Using the optimizers of Adam or Adadelta has no obvious effects on the performance, which only causes a difference around 2.5% in the accuracy. But both the accuracy and loss curves fluctuate more in the latter part of training with Adam than with Adadelta. In addition, the training accuracy and loss converged faster at the beginning with Adadelta.

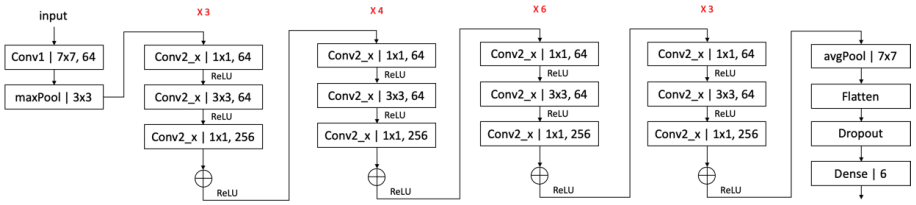

Fig. 4. Structure of the plain ResNet50 model.

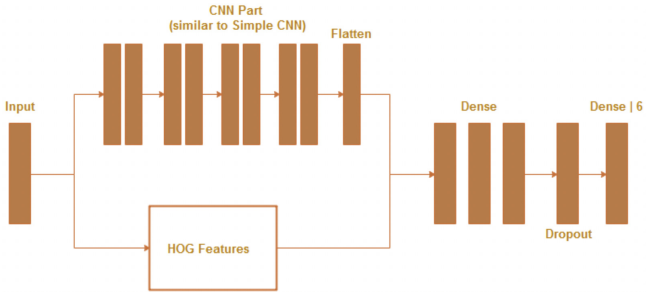

Fig. 5. Structure of the hybrid model.

TABLE II
PERFORMANCE OF THE RESNET50 MODEL.

Optimizers	Training Accuracy	Test Accuracy
Adam	96.91%	51.67%
Adadelta	99.27 %	95.35%

The confusion matrix in Fig. 7 shows that the simple CNN architecture is successful with almost all classes except plastic. There is a large probability that the model may mistake plastic garbage with glass and paper, or mistake metal garbage with glass.

3) ResNet50: Table II shows classification performance of the ResNet50 model. The ResNet50 model achieves a training accuracy of 99% and a test accuracy of 95% with Adadelta, while the test accuracy just reached 51% and the overfitting is obvious with Adam.

Fig. 8 shows the evolutions of training/test accuracies and training/test losses as the number of epochs increases. The training accuracy along with the training loss converged more efficient with the Adadelta optimizer. Furthermore, the experiments with Adam tend to have the overfitting problem, as the validation curves fluctuate much and have no signs of convergence.

4) Plain Network for ResNet50: Table II shows classification performance of the plain network of ResNet50. The plain network has worse performance than the others. The training accuracy and test accuracy both reached 76%. These results further confirm the importance of skip connection.

5) HOG+CNN model: Table II shows classification performance of the HOG+CNN approach. The combination of HOG features and CNN features yields good performance in the experiments. The test accuracy of this model is slightly higher than the results of the simple CNN architecture. After 40 epochs, the training accuracy reached 89% and validation accuracy over 93% with the Adadelta optimizer.

TABLE III
PERFORMANCE OF THE PLAIN NETWORK OF RESNET50.

Optimizers	Training Accuracy	Test Accuracy
Adam	45.14%	35.30%
Adadelta	76.41 %	76.93%

TABLE IV
PERFORMANCE OF THE HOG+CNN APPROACH.

Optimizers	Training Accuracy	Test Accuracy
Adam	81.98%	82.19%
Adadelta	89.52 %	93.56%

C. Performance Comparison

Garbage classification is a Kaggle challenge, and many researchers have submitted their results. To understand the position of the proposed methods, we compare our performance with that announced on the website.

Table V shows the comparison of this project with other current trials on garbage classification. Please notice that all the other attempts run the models on the original dataset with 2,527 images. The publisher of the dataset construct an SVM classifier based on scale-invariant feature transform (SIFT) features, which achieves a test accuracy of 63% [8]. Another project also uses data augmentation with DenseNet121 and achieves a test accuracy of 95% after 200 epochs of training and 10 epochs of fine tuning. The RecycleNet uses the architecture of DenseNet with an alternation of the skip connections and achieves a test accuracy of 81% [9] after 200 epochs. The top-1 model on Kaggle adopts MobileNetV2 with the sigmoid activation function and binary cross entropy loss function [10]. The top-2 model adopts a simple CNN architecture similar to ours and also reaches a high accuracy after 100 epochs [11].

The values in the top half of Table V cannot be directly compared with the values in the bottom half because the evaluated data are not exactly the same. However, we still can make two observations. First, the developed ResNet50 model achieves competitive performance with the state of the art. Second, data augmentation is important. For example, the ResNet50 model improves the classification accuracy from 91.40% to 95.35% if the data are augmented.

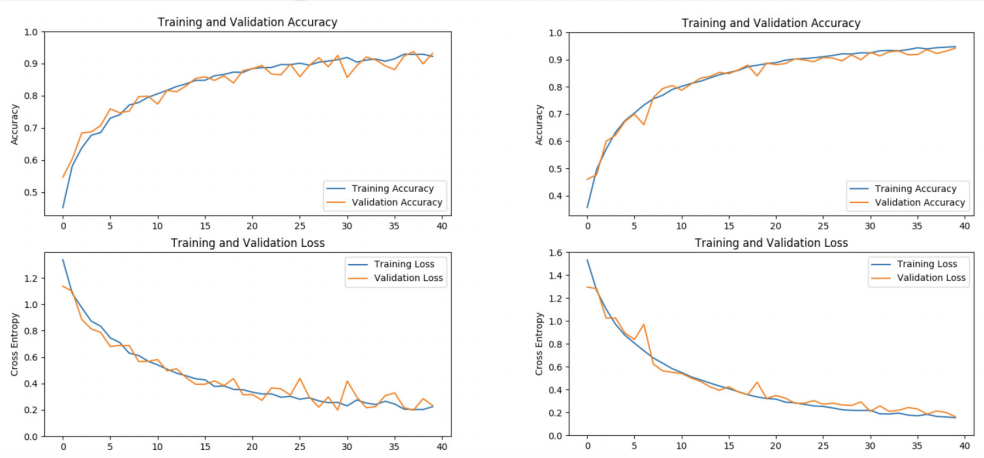

Fig. 6. Evaluations of accuracy and loss based on the simple CNN model. Left column: training by the Adam optimizer; right column: training by the Adadelta optimizer.

TABLE V
PERFORMANCE COMPARISON BETWEEN DIFFERENT APPROACHES.

Approach	Test Accuracy	Notes	Epochs
SVM+HOG	47.25%	9,095 train img, 1,013 test img	–
	23.51%	2,276 train img, 251 test img	–
Simple CNN	93.75%	9,095 train img, 1,013 test img	40
	79.49%	2,276 train img, 251 test img	40
ResNet50	**95.35%**	9,095 train img, 1,013 test img	40
	91.40%	2,276 train img, 251 test img	40
HOG+CNN	93.56%	9,095 train img, 1,013 test img	40
	81.53%	2,276 train img, 251 test img	40
SIFT + SVM [8]	63%	1,769 train img, 758 test img	–
DenseNet121 [9]	95%	Vertical and horizontal flip, 15-degree rotation with fine tuning	200+10
RecycleNet [9]	81%	Vertical and horizontal flip, 15-degree rotation	200
MobileNetV2 (top 1 at Kaggle) [10]	94.89%	2276 train img, 251 val. img with fine tuning	10+10
CNN (top 2 at Kaggle) [11]	84%	2276 train img, 251 val. img	100

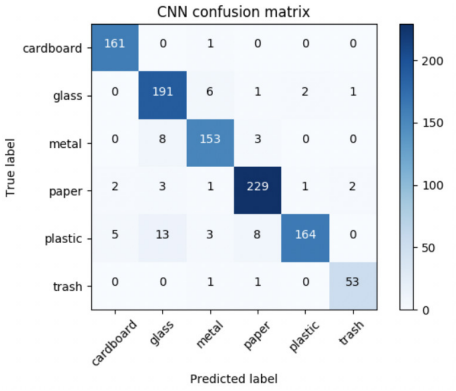

Fig. 7. Confusion matrix of simple CNN architecture with Adadelta.

D. Discussion

Fig. 9 shows evaluations of accuracy and loss based on different approaches. As can be seen, the ResNet50 model is efficiently converged in terms of training accuracy and training loss. Because of the pre-trained weights initialization, ResNet50 begins the training with a high accuracy and a low loss. The simple CNN model converges smoothly and could reach high accuracy for the problem to some extent. Although the plain network of ResNet50 has more conv. layers than the simple CNN, it works inefficiently and seems to need more time to train and to achieve better performance. The reason may lie in the redundant 1×1 conv. layers in the structure, which is unproductive to the training process.

The combination of HOG features and CNN achieves worse performance in terms of training accuracy and training loss, comparing to the simple CNN model. However, the testing accuracy of this model is as good as the simple CNN as they have the same order of magnitude of parameters. To some extent performance of the HOG+CNN model is even better than the simple CNN given less data input.

Although the Adam optimizer is seen as an improved optimizer than Adadelta, it doesn't train the models better according to our empirical results. Learning rate of the Adam optimizer may be too small in latter part of training to converge well [12], which could also explain the fluctuations of the accuracy and loss curves in the last 10 epochs.

From the performance comparison (Table V) we could draw the conclusion that expanding the database and increasing epochs can both improve the accuracy. Data source acquisition

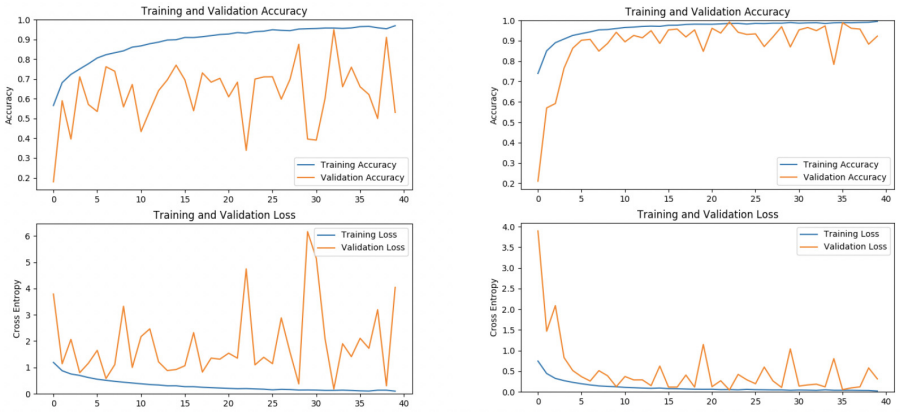

Fig. 8. Evaluations of accuracy and loss based on the ResNet50 model. Left column: training by the Adam optimizer; right column: training by the Adadelta optimizer.

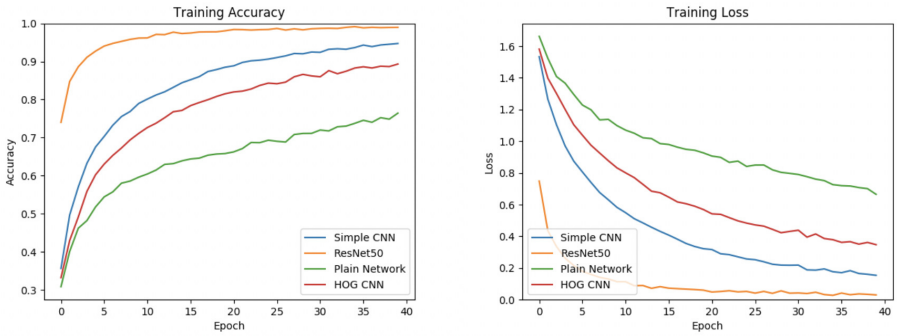

Fig. 9. Evaluations of accuracy and loss based on different approaches.

and completion is thus important for practical garbage classification, which could lead to a more rapid growth in accuracy and higher precision in prediction.

V. Conclusion

From the results of this study we can see, the problem of garbage image classification can be solved with deep learning techniques at a quite high accuracy. The combination of specific features with CNNs or even other transferring models might be an efficient approach to do the classification. However, it is unrealistic to get a picture of an object on the clean background each time when people classify the garbage. Due to the large variety of garbage categories in real life, the model still needs a larger and more precisely classified data source taken in more complicated situations.

Acknowledgment

This work was partially supported by the Ministry of Science and Technology, Taiwan, under the grant 108-2221-E-006-227-MY3, 107-2221-E-006-239-MY2, 107-2923-E-194-003-MY3, 107-2627-H-155-001, and 107-2218-E-002-055.

References

[1] Silpa Kaza, Lisa Yao, Perinaz Bhada-Tata, and Frank Van Woerden, *What a Waste 2.0: A Global Snapshot of Solid Waste Management to 2050*, World Bank, 2018.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2012.

[3] Yann LeCun, Koray Kavukcuoglu, and Clement Farabet, "Convolutional networks and applications in vision," in *Proceedings of IEEE International Symposium on Circuits and Systems*, 2010.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[5] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of International Conference on Learning Representations*, 2015.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[7] Chih-Chung Chang and Chih-Jen Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, 2011.

[8] Mindy Yang and Gary Thung, "Classification of trash for recyclability status," *Final Project Report, CS229, Stanford University*, 2016.

[9] Cenk Bircanoğlu, Meltem Atay, Fuat Beşer, Özgün Genç, and Merve Ayyüce Kızrak, "Recyclenet: Intelligent waste sorting using deep neural networks," in *Proceedings of Innovations in Intelligent Systems and Applications*, 2018.

[10] "Transfer learning using mobilenet," in *https://www.kaggle.com/hamzakhan/transfer-learning-using-mobilenet*, 2019.

[11] "Using cnn [test accuracy- 84%]," in *https://www.kaggle.com/pranavmicro7/using-cnn-test-accuracy-84*, 2019.

[12] Nitish Shirish Keskar and Richard Socher, "Improving generalization performance by switching from adam to sgd," in *https://arxiv.org/abs/1712.07628*, 2017.