A Deep Convolutional Neural Network for classifying waste containers as full or not full

Bania J. Fonseca
Department of Informatics Engineering
Lúrio University
Pemba, Mozambique
bfonseca@unilurio.ac.mz

Felermino D. M. A. Ali
Department of Informatics Engineering
Lúrio University
Pemba, Mozambique
felermino.ali@unilurio.ac.mz

Saide M. Saide
Department of Informatics Engineering
Lúrio University
Pemba, Mozambique
saide.saide@unilurio.ac.mz

Abstract— **There is a common understanding that cleanliness is somehow proportional to the economic development of a country. Thus, in order to become clean, a country needs to have an efficient garbage monitoring system. One important component of such a system is garbage collection time because if we delay emptying the bins, the trash ends up to putting public health at risk. This paper is about creating a Deep Convolutional Neural Networks (DCNNs) based model for classifying a waste container as full or not, so that can be later on used by real-time garbage monitoring systems to process images acquired by cameras installed nearby the trash bins or smartphones. To achieve this, we trained and tested different well-known DCNNs architectures, namely, ResNet34, ResNet50, Inception_v4 and DarkNet53. The models were trained and tested using Repeated K-Fold Cross-Validation, running 5-Fold Cross-Validation 6 times. The results have showed that Inception_v4 outperformed the other models, with near-perfect results: PR-AUC = 0.994, F1 = 0.988, Precision = 0.989, Recall = 0.987 and ACC = 0.987. With these results can be said: a high Precision DCNNs based model was built.**

Keywords— Deep Learning, Image Classification, Pattern Recognition, Artificial Neural Networks.

I. INTRODUCTION

Waste has become a major problem around the world due to industrialization, excessive consumption, and lack of effective and efficient waste management. This problem is so serious that poses a threat not only to public health but also impacts negatively to the environment [1]. In underdeveloped countries like Mozambique, where the waste management is poor it is common to see waste containers full for quite a number of days making the place filthy with an unpleasant view and smell, therefore putting surrounding people at risk. So, on-time waste collection response is important and one way to achieve that is to have a real-time alert system that triggers as soon as the container became full. Instrumentation can be one option to choose when implementing such systems. Instrumentation is the delivery of real-world data from sensors (smartphones, ultrasonic sensors, surveillance cameras, RFID detectors, etc.) in real time [2]. Many approaches have been proposed to instrument waste container, for instance, some approaches have used embedded systems combined with ultrasonic and infrared sensors [3] and [4]. The drawback of this approaches is that they require do install sensor on the waste repository, therefore, introducing flexibility risk, cost problems and a high environmental impact [5]. This study presents a different approach which make use of Deep Convolutional Neural Networks (DCNNs) to classify the state of waste container: "Whether they contain enough waste to collect or not," from images taken by citizen's own smartphones or cameras installed near waste container. Effective waste container instrumentation can significantly influence the waste collection process, and a cost-effective solution would be a participatory collaboration with citizens [6] who would use their smartphones to capture images of waste container that would be automatically sent to a server running a prediction model built using DCNNs to classify the state of waste container. However, a question arises: "Can we achieve high Precision in classifying the state of waste container in the outdoor through the use of DCNNs?

This paper is structured as follow: section II Related works, section III methodology, IV Results and Discussion, Conclusion, Acknowledgement and References.

II. RELATED WORKS

Deep Convolutional Neural Networks (DCNNs) are the predominant types of neural networks used to multidimensional signal processing. The term deep refers generally to networks with multiple convolution layers, and deep learning refers to training methodologies of these networks [7].

DCNNs are currently the approach of choice to deal with complex image recognition tasks and other areas including speech recognition, semantic image segmentation and natural language processing [7]. DCNNs architecture consists of: convolutions, activation and pooling. A DCNNs is composed of several stages interconnected in series. The Fig. 1 shows the basic components of a DCNN stage.

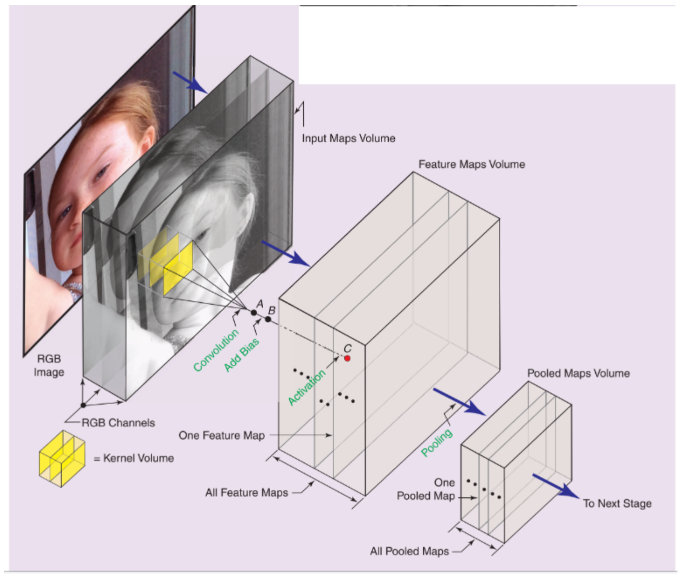

Fig. 1. Basic components of a DCNN stage [7]

The process of training a neural network is a problem of optimization; whose purpose is to adjust all parameters of the network when a misclassification is made so that the error in the output is minimized. This is done using gradient descent:

$$w_{ij}(l) = w_{ij}(l) - \propto \frac{\partial E}{\partial w_{ij}(l)} \qquad (1)$$

$\propto$: learning rate
$w_{ij}(l)$: parameter or weight
l : neural network layer

This study sought to explore the possibility of DCNNs achieving high Precision in classifying the state of the waste container in an outdoor environment. To the best of our knowledge, there not studies that explores DCNNs for classifying if a waste container is full or not. Most studies touch the issue of segregating waste inside the container in classes such as paper, glass, metal, plastic, etc. [1][8][9]. DCNNs was used in [5] to identify waste container found in the outdoor. Other studies explored the possibility of determining the state of waste container using embedded systems equipped with ultrasonic and infrared sensors [3][4]. This is a desirable solution, however, this technology needs to be coupled to the waste container. This introduces flexibility, risk and cost issues and a high environmental impact [5] These were the closest studies to ours, but none of these studies have explored the possibility of DCNNs achieving high Precision in classifying the state of the waste container in the outdoor.

.

III. METHODOLOGY

A. Building the dataset

The dataset was built from images captured from waste containers available at different location in some Mozambique cities. The dataset was initially composed by 65 images of waste container labeled as full and 75 labeled as not full. Since the size of the dataset was small and not enough to train the selected DCNNs, the following three augmentation techniques have been applied in order to avoid overfitting problem: blur, flip and random noise. The applied data augmentation technique made a slightly change on the images, so the augmented data did not differ too much to the original data, this can be seen at the Fig. 2. The final dataset is composed by 260 images of waste container labeled as full and 300 labeled as not full. All images in the dataset were automatically resized to 256x256 pixels.

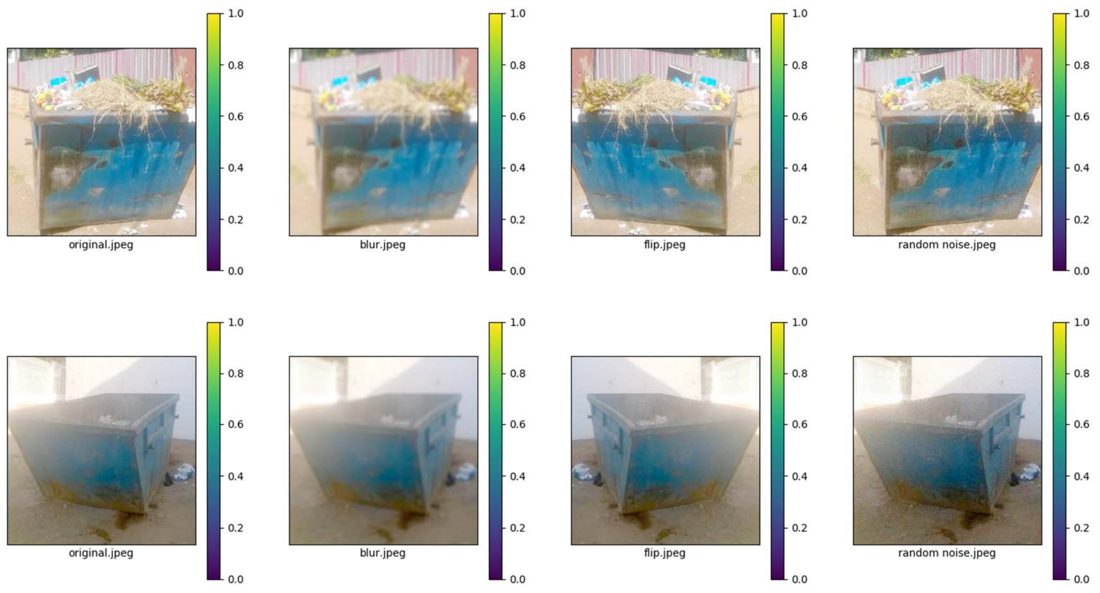

Fig. 2. Some images of the built dataset with augmented data. The first row is composed by container images labeled as full and the second row is composed by container images labeled as not full.

B. *Building the models*

Tensorflow with keras API was used to build models by taking into account the following DCNNs architecture: ResNet34, ResNet50, Inception_v4 and DarkNet53. The RetinaNet detention model uses a Feature Pyramid Network (FPN) backbone on top of a feed-forward ResNet architecture [10]. The Yolo_v3 detention model uses DarkNet53 for performing feature extraction [11].

C. *Training and Testing*

One of the problem with evaluating a model is that it may demonstrate adequate prediction capability on the training data, but might fail to predict future unseen data and cross-validation is a procedure for estimating the generalization performance in this context [12].

The procedure chosen to train and test the models was Repeated K-Fold Cross-Validation. "In k-fold cross-validation the data is first partitioned into k equally (or nearly equally) sized segments or folds. Subsequently k iterations of training and validation are performed such that within each iteration a different fold of the data is held-out for validation while the remaining k-1 folds are used for learning" [12]. As in k-fold cross-validation, only k estimates are obtained, a commonly used method to increase the number of estimates and to obtain less biased performance estimation is to run k-fold cross-validation multiple times. The data is reshuffled and re-stratified before each round [12]. In this study the 5-fold cross-validation was run 6 times, which gave 30 points of estimation for each metric, namely: PR-AUC, F1, PRECISION, RECALL and ACCURACY [11].

The mean and sample standard deviation for each metric were calculated, as the points of estimation is equal to 30 we built a confidence interval (CI) with a level of confidence ***c* = 0.95.** We used *ADAM* [13] optimizer with learning rate of 1e-4 during all training process and sparse_categorical _crossentropy as loss function.

D. *Finding the correlation between depth and ACC*

"The most straightforward way of improving the performance of deep neural networks is by increasing their size. This includes both increasing the depth – the number of network levels – as well as its width: the number of units at each level" [14]. In order to find out if this happen on the chosen models, the correlation between the model depth and ACC was calculated.

IV. RESULTS AND DISCUSSION

All the experiments of this study were performed using TensorFlow version 1.13.1 with Keras API on a hardware with 8 GB of RAM and 7th generation i7 dual core processor with 2.90 GHZ. The sklearn library was used to compute the performance of the models. The performances of the models are shown in Table I.

Table 1. Resumed performances of the models

Models	PR-AUC	F1	PRECISION	RECALL	ACC	DEPTH
Inception_v4	**0.994 +/- 0.002**	**0.988 +/- 0.004**	**0.989 +/- 0.005**	**0.987 +/- 0.005**	**0.987 +/- 0.004**	**75**
ResNet50	0.989 +/- 0.003	0.976 +/- 0.005	0.975 +/- 0.009	0.978 +/- 0.007	0.974 +/- 0.006	**53**
Darknet53	0.99 +/- 0.003	0.976 +/- 0.006	0.972 +/- 0.009	0.98 +/- 0.007	0.974 +/- 0.007	**51**
ResNet34	0.988 +/- 0.003	0.975 +/- 0.004	0.968 +/- 0.009	0.976 +/- 0.006	0.969 +/- 0.005	**37**

Overall, the results were good, because the smallest estimated point is 0.968, which is the Precision of ResNet34. As was observed by [14] "The most straightforward way of improving the performance of deep neural networks is by increasing their size. This includes both increasing the depth – the number of network levels – as well as its width: the number of units at each level", and that was verified at this study. As we can see at Fig. 3, with a pearson correlation coefficient ***r* = 0.986**, the correlation between the model depth and the ACC is very strong positive, roughly speaking deeper models have tended to have higher ACC.

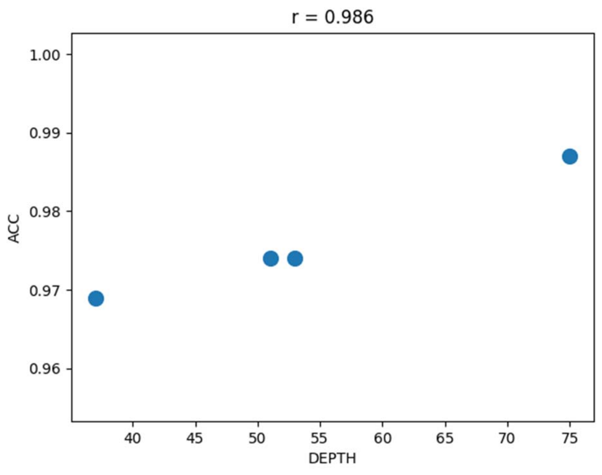

Fig. 3. Very strong positive correlation between the model depth and ACC.

A. Inception_v4

The Table 2 show more details about the performance of Inception_v4, the standard deviation calculated from estimates obtained by running 5-Fold Cross-Validation 6 times is nearly zero, which mean, there is just slightly deference between the obtained estimates or roughly speaking the obtained estimates at each time of test process was nearly equal, which make the model more consistent.

Table 2. The performance of Inception_v4. Margin of Error, ***E***. Sample standard deviation, ***S.*** Confidence Interval, **CI.**

Metrics	mean	*S*	*E*	CI
PR-AUC	0.994	0.007	0.002	0.994 +/- 0.002
F1	0.988	0.012	0.004	0.988 +/- 0.004
PRECISION	0.989	0.015	0.005	0.989 +/- 0.005
RECALL	0.987	0.014	0.005	0.987 +/- 0.005
ACC	0.987	0.012	0.004	0.987 +/- 0.004

With Precision of 0.989, and Recall of 0.987, the model had on average the absence of false positive (false alarms) and presence of only one false negative. It is important to point out that, the Inception_v4 had less parameters than DarkNet53 and ResNet50. Also, differently from DarkNet53 and ResNet50 Inception_v4 do not use Residual block as building blocks [15], instead it uses Inception block as building blocks [16], which is much deeper, sparse and with very variability in the kernels size, leading to good results. This supports the idea that most of the progress in deep learning is not just on the result of more powerful hardware, larger datasets and bigger models, but mainly a consequence of new ideas, algorithms and improved network architectures [14].

B. ResNet50

The Table 3 show more details about the performance of ResNet50, the standard deviation is nearly zero.

Table 3. The performance of ResNet50

Metrics	mean	*S*	*E*	CI
PR-AUC	0.989	0.009	0.003	0.989 +/- 0.003
F1	0.976	0.014	0.005	0.976 +/- 0.005
PRECISION	0.975	0.024	0.009	0.975 +/- 0.009
RECALL	0.978	0.020	0.007	0.978 +/- 0.007
ACC	0.974	0.016	0.006	0.974 +/- 0.006

With Precision of 0.975, and Recall of 0.978, the model had on average the presence of only one false positive (false alarms) and presence of only one false negative. We made a slightly change on the ResNet50 architecture, removing all Batch Normalization (BN), why we did that? Neural networks suffer with covariate shift problem, and this happen when the distribution of each layer's inputs change constantly resulted by the change of parameters of previous layers and the change of distribution of the first layer's inputs during training [17] and the application of BN not only addresses this problem, but greatly enhance the speed of neural network training [17]. But this was not verified on our training process, as can be seen at Table 4, training ResNet34 with BN required more time and gave bad results, so we removed BN in ResNet34, ResNet50 and DarkNet53 because was slowing down the training process and reducing a little the models performances. This study did not go in depth to discover why Batch Normalization was not speeding our training and leading to better results.

Table 4 ResNet34 with BN vs. ResNet34 without BN

ResNet34 with BN					**ResNet34 without BN**			
Epoch	time (s)	train ACC	test ACC		Epoch	time (s)	train ACC	test ACC
1	145	0.683	0.464		1	97	0.551	0.633
2	132	0.928	0.464		2	96	0.647	0.66
3	136	0.986	0.464	vs.	3	96	0.77	0.714
4	133	0.995	0.464		4	95	0.825	0.714
5	137	0.991	0.464		5	97	0.819	0.83
6	134	1	0.464		6	96	0.897	0.83

C. DarkNet53

The Table 5 show more details about the performance of DarkNet53, the standard deviation is nearly zero.

Table 5. The performance of DarkNet53

Metrics	mean	*S*	*E*	CI
PR-AUC	0.990	0.009	0.003	0.99 +/- 0.003
F1	0.976	0.018	0.006	0.976 +/- 0.006
PRECISION	0.972	0.024	0.009	0.972 +/- 0.009
RECALL	0.980	0.020	0.007	0.98 +/- 0.007
ACC	0.974	0.019	0.007	0.974 +/- 0.007

With Precision of 0.972, and Recall of 0.98, the model had on average the presence of only one false positive (false alarms) and presence of only one false negative. We made a slightly change on the DarkNet53 architecture: we applied the approach used at ResNet architecture, replacing the first two convolutional layer of DarkNet53, by one convolution layer with 7x7 kernel size followed by a 3x3 max pool, which lead to good results.

D. ResNet34

The Table 6 show more details about the performance of ResNet34, the standard deviation is nearly zero.

Table 6. The performance of ResNet34

Metrics	mean	*S*	*E*	CI
PR-AUC	0.988	0.009	0.003	0.988 +/- 0.003
F1	0.975	0.013	0.004	0.975 +/- 0.004
PRECISION	0.968	0.026	0.009	0.968 +/- 0.009
RECALL	0.976	0.018	0.006	0.976 +/- 0.006
ACC	0.969	0.015	0.005	0.969 +/- 0.005

With Precision of 0.968, and Recall of 0.976, the model had on average the presence of only two false positive (false alarms) and presence of only one false negative. We made a slightly change on the ResNet34 architecture we used a residual block with 1x1 kernel size followed by 3x3 kernel size, instead of residual block with only 3x3 kernel size.

CONCLUSION

In this study, the experiments were performed on known DCNNs. As a result of these experiments, all implemented models achieved at least 0.969 of Accuracy, 0.968 of Precision and 0.976 of Recall. The highest scores were achieved by Inception_v4 with 0.987 of Accuracy, 0.989 of Precision and 0.987 of Recall, which has less parameter number than ResNet50 and Darknet53.

In the light of the experiments performed, it can be said that: high Precision can be achieved in the classification of the state of waste container that are in the outdoor through the use of DCNNs. Since those models can classify a waste container as “full or not”, real-time waste monitoring systems can use them to determine when to empty the waste containers. In the next study, the size of the dataset will be increased to obtain more successful results and even less biased estimation.

ACKNOWLEDGEMENT

This paper would not have been possible without God and the support of a number of people, all of whom we wish to acknowledge with gratitude for their help and support.

We are also grateful to the Lúrio University – Faculty of Engineering and the National Research Fund of Mozambique (*Fundo Nacional de Investigação - FNI*), for their financial support.

REFERENCES

[1] B. S. Costa *et al.*, “Artificial Intelligence in Automated Sorting in Trash Recycling,” no. October, pp. 198–205, 2018.

[2] C. Harrison *et al.*, “Foundations for Smarter Cities,” *IBM J. Res. Dev.*, vol. 54, no. 4, pp. 1–16, 2011.

[3] A. Anitha, “Garbage monitoring system using IoT,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 263, no. 4, 2017.

[4] C. J. Baby, H. Singh, A. Srivastava, R. Dhawan, and P. Mahalakshmi, *Smart Bin: An Intelligent Waste Alert and Prediction System Using Machine Learning Approach*. 2017.

[5] G. P. D. Valente M., Silva H., Caldeira J. M. L. P., Soares V. N. G. J., “Detection of Waste Containers Using Computer Vision,” 2019.

[6] S. Castillo, N. M., Vosloo, “hearScreen^{TM} Case study by UNESCO-Pearson Initiative for Literacy,” 2015.

[7] R. C. Gonzalez, “Deep Convolutional Neural Networks,” *IEEE SIgnal ProcESSIng MagazInE*, 2018.

[8] H. M. Aral R. A., Keskin S. R., Mahmut K., *Classification of TrashNet Dataset Based on Deep Learning Models*. 2018.

[9] K. S. Hulyalkar S., Deshpande R., Makode K., “IMPLEMENTATION OF SMARTBIN USING CONVOLUTIONAL NEURAL NETWORKS,” *Int. Res. J. Eng. Technol.*, 2018.

[10] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal Loss for Dense Object Detection,” 2017.

[11] J. Redmon and A. Farhadi, “YOLO v.3,” pp. 1–6, 2018.

[12] P. Refaeilzadeh, L. Tang, and H. Liu, “Cross-Validation,” 2009.

[13] S. Indolia, A. Kumar Goswami, S. Mishra, and P. Asopa, “Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach,” 2018.

[14] C. Szegedy *et al.*, *Going Deeper with Convolutions*. 2015.

[15] B. Li, “An Improved ResNet Based on the Adjustable Shortcut Connections,” 2018.

[16] C. Szegedy, S. Ioffe, and V. Vanhoucke, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” 2016.

[17] L. Chen, H. Fei, Y. Xiao, J. He, and H. Li, “Why batch normalization works? a buckling perspective,” in *2017 IEEE International Conference on Information and Automation, ICIA 2017*, 2017, pp. 1184–1189.