Convolutional Neural Network Approaches to Sort Garbage

Richard Nai

Abstract Proper garbage disposal is a growing problem in this day and age. Proper recycling of recyclable waste can bring about major health and economic benefits. In this work, computer vision and convolutional neural network (CNN) techniques will be applied to differentiate which type of garbage is in an RGB image. The TrashNet dataset is applied to train the networks. Infrastructure is also provided to train new architectures on the TrashNet dataset. A ROS2 enviornment and standalone C++ application are then presented that leverage the trained networks to perform inference on unseen images of garbage taken in domestic settings.

Keywords Convolutional Neural Network · Image Classification · Computer Vision

1 Introduction

This project is motivated by the growing amount of garbage in the world. In many urban cities, streets are accumulating large amounts of garbage. Not only does this ruin the visual appeal of the landscape, but it can cause environmental and health hazards if not dealt with in time [10]. It is estimated that at least 1 million birds, one million mammals and an inestimable number of fish are killed by floating aquatic waste alone each year [11]. Proper recycling of recyclable materials is crucial for a sustainable future. Correct separation and sorting of garbage is crucial and this process is currently done manually [13]. Improper recycling causes higher waste accumulation and reduces the amount of re-use of disposed products, which has damaging environmental impact. Proper recycling presents both environmental benefits as a result of reduced waste and economic benefits as a result of increased waste disposal plant efficiency [12, 13].

The key contributions of this paper are the creation and training of Resnet18 and ResNet152 networks to properly identify recyclable materials. They have achieved accuracies of 92.52% and 95.67% respectively on the TrashNet image set. Additionally, the framework used to train them is flexible and can quickly train other CNN architectures on the TrashNet dataset and has been made publicly available on GitHub. Furthermore, the trained instances were then used to create a ROS2 program and standalone C++ program, both of which could parse video streams and correctly predict garbage in video streams.

This paper is organized as follows: Section 2 presents related work in garbage identification, classification, and construction of autonomous systems to deal with them. Section 3 describes our proposed CNN model to perform garbage identification and classification while Section 4, the experiments and results. Finally, conclusions, limitations, and discussions of future work are given in Section 5.

2 Related Work

This paper will use TrashNet to train and validate CNNs on. TrashNet is a dataset of around 2500 images, published by Yang and Thung in their 2016 paper Classification of Trash for Recyclability Status [14]. It was the first image dataset specific to trash classification, and contained 6 classes: paper, glass, plastic, metal, cardboard and unrecyclabe trash. Pictures of the dataset can be seen in figure 1. This dataset has been used by Aral et al. in their 2018 paper Classification of TrashNet Dataset Based on Deep Learning Models. In that paper, various deep learning models were used, namely Xcep-

Richard Nai
Technical University of Munich
E-mail: richard.nai@tum.de

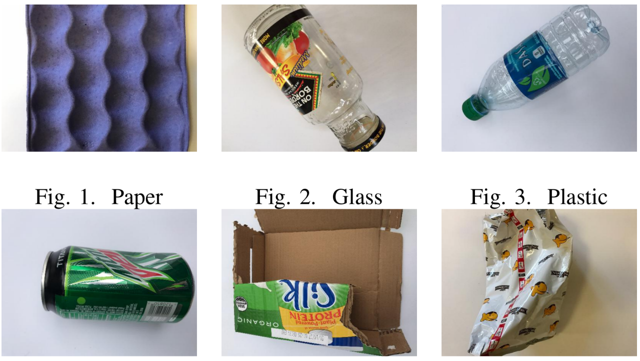


Fig. 1. Paper

Fig. 2. Glass

Fig. 3. Plastic

Fig. 4. Metal

Fig. 5. Cardboard

Fig. 6. Trash

Fig. 1: TrashNet Classes [14]

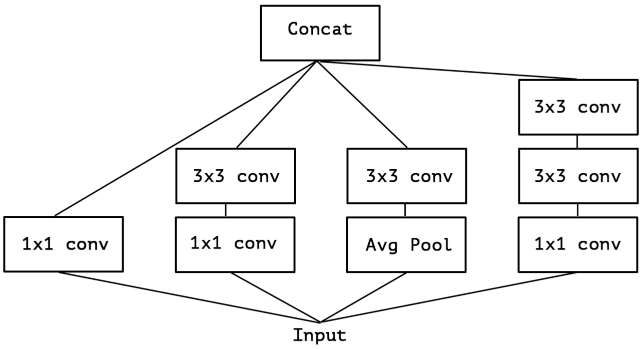


Fig. 2: InceptionNet V3 [6]

tion, MobileNet, Densenet121, DenseNet169 and InceptionResnetV2 [4]. Xception is an architecture based on depthwise separable convolutional layers and is based of Google's InceptionNet. InceptionNet learns cross-channel and spacial correlations faster than a standard convolutional layer by performing a series of convolutions that look at them independently. An input image is convolved using filters with different sizes and then put back together. [6]. XceptionNet utilizes this to map the cross-channel and spatial correlations completely separately [6]. A graphical representation of InceptionNet can be see in figure 2 and a graphical representation of XceptionNet can be see in figure 3. MobileNet is another model developed by the Google that is comprised of a full convolution in the first layer, followed by in-depth separable convolutions similar to InceptionNet [4]. DenseNets, or Densly Connected Convolutional Networks, are neural network structures that contain very efficient convolutional neural structures and have short connections between input and output layers [4]. In a DenseNet, each layer is an input to every subsequent layer [9]. Lastly, InceptionResnetV2 is a combination of the techniques used in InceptionNet and ResNet [4].

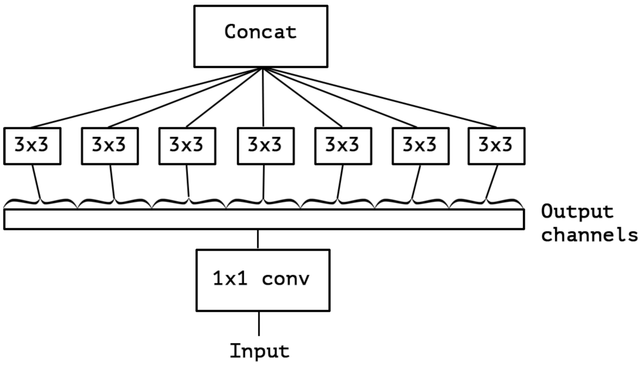


Fig. 3: XceptionNet Architecture [6]

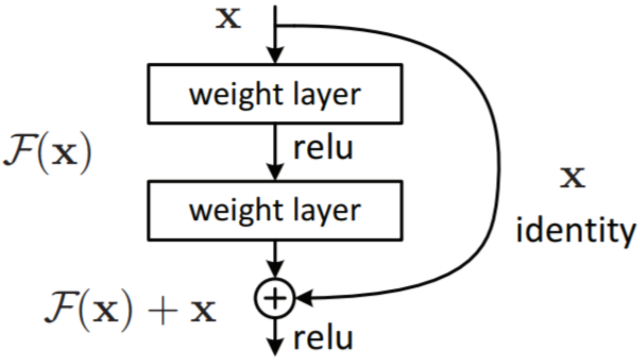
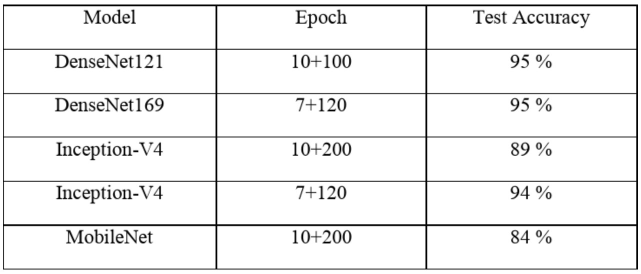


Fig. 4: Residual Block [8]



Model	Epoch	Test Accuracy
DenseNet121	10+100	95 %
DenseNet169	7+120	95 %
Inception-V4	10+200	89 %
Inception-V4	7+120	94 %
MobileNet	10+200	84 %

Fig. 5: Aral et al.'s Model Accuracies [6]

Aral *et. al*'s results after training and fine tuning are shown in figure 5. Thus, it can be seen that both DenseNet architectures performed the best with 95% classification accuracy.

In this paper, the ResNet architecure will be used due to its lower memory usage [2]. ResNet is an architecture where the input to a layer is connected to the output of the layer via an identity function to allow for better gradient flow. These happen in special blocks called Residual blocks, which is where the name of ResNet comes from [8]. An example of a residual block can be seen in figure 4.

Some other noteworthy garbage recognition works are as follows. Auto-Trash is a smart garbage bin that uses a camera and Raspberry Pi to perform image classification using TensorFlow to separate compostables from other waste [1]. Unfortunately, no public dataset was provided. Mittal et al. present their garbage identification system in their 2016 paper: SpotGarbage: Smartphone App to Detect Garbage Using Deep Learning [10]. They present a dataset of images of garbage in the wild that are annotated with garbage severity and perceived biodegradability called Garbage In Images (GINI). GINI was created by using the Bing Image Search API to crawl the internet for garbage related images. This resulted in collection of 2561 images. This was then analyzed by manually pruning out irrelevant images (such as clipart of garbage) and annotating if and where there the garbage was in the image, as well as perceived severity and biodegradability [10]. Due to the large congolmerated garbage and lack of recyclability labels, the GINI dataset was not suitable for this project. Mittal et al. also developed a fully convolutional network called GarbNet, which is based off AlexNet, that achieves an 87.69% accuracy on the GINI dataset. GarbNet is used by a mobile app called SpotGarbage to empower users to notify their governments of locations of garbage [10]. Bai et al. present a robotic deep learning system which is capable of autonomously collecting garbage in their 2018 paper: Deep Learning Based Robot for Automatically Picking Up Garbage on the Grass [5]. Their system utilizes an RGB camera mounted on a robot to identify and locate objects in a grassy field, and then to classify if those objects are garbage. The garbage detection workflow is as follows: a SegNet based network is used to perform image segmentation on an input image from the camera. The network will then return the corresponding ground segmentation and draw bounding boxes around any objects detected. If there are objects detected in the image, the robot will move closer to get more detail. A ResNet based classifier will then be used to determine if the object is garbage [5]. Unfortunately, no public datasets were given from that project. Niu *et al.* present a unmanned aerial vehicle (UAV) based garbage identification system in their 2019 paper SuperDock: A Deep Learning-Based Automated Floating Trash Monitoring System [11]. In their implementation, an UAV with a camera and TX2 used the YOLOv3 algorithm to perform trash classification and localization. The network was first pretrained with ImageNet and then fine tuned with a custom floating trash dataset. After finetuning, their modified YOLOV3 model was able to achieve 81.2% accuracy with an average processing time of 0.038 seconds on a desktop computer with an i7 CPU and Nvidia GTX 1060 GPU [11].

3 Methodology

In this implementation, the ResNet [8] architecture was used, due to its integration into PyTorch and quick convergence during training. PyTorch allows out of the box creation of ResNet18 and ResNet152 instances that have been pretrained on ImageNet [7]. Those networks were used as a starting point, and then the last level fully connected layers were then removed and replaced with a fully connected layer with 6 output neurons, to match the number of classes in TrashNet. The ResNet18 network architectures can be seen in figure 6. The input data was first passed through a 2d convolutional layer, and then normalized, given nonlinearity and pooled. The resultant data was then passed through 4 sequential nodes with residual blocks. Finally, the output was then pooled again and passed through a fully connected layer, which computed the logits of each class. During training, cross entropy loss was then applied on the logits to compute the loss and backpropagate the network. The ResNet152 architecture followed the same structure, but had more weights in each subnode.

The dataset was split into a training, validation, and test dataset with percentages of 80%, 10% and 10% respectively. PyTorch's random seed was set to 0 to ensure the same split across different runs. Furthermore, the dataset was augmented by applying random horizontal and flips.

The training methodology is as follows. The Adam optimizer was used with initial learning rate of 0.001 and batch size 8. The network would then be trained for 1 epoch, and the gradients for all layers except the last fully connected layer would be frozen. The learning rate and batch size were lowered as the model converged. Once the model approached 80% accuracy, all the layers would be unfrozen, and the network would be trained until validation accuracy no longer increased. For both networks, the initial batch size was 8 and the initial learning rate was 0.001. The beta and epsilon parameters for the Adam optimizer were left at PyTorch's default values of 0.9, 0.999, and 1e-8 respectively. Weight decay was also not used, and the learning rate was manually adjusted between epochs. This was to ensure a reasonable learning rate was being used, and to remove an extra hyper parameter to tune, although a good weight decay parameter would achieve similar results.

Finally, the PyTorch models were converted into C++ models using PyTorch's TorchScript [3] and then used to make a ROS2 node and standalone C++ application to perform inference on input images. Using PyTorch's C++ bindings, serialized models could be loaded and used to perform inference without the need

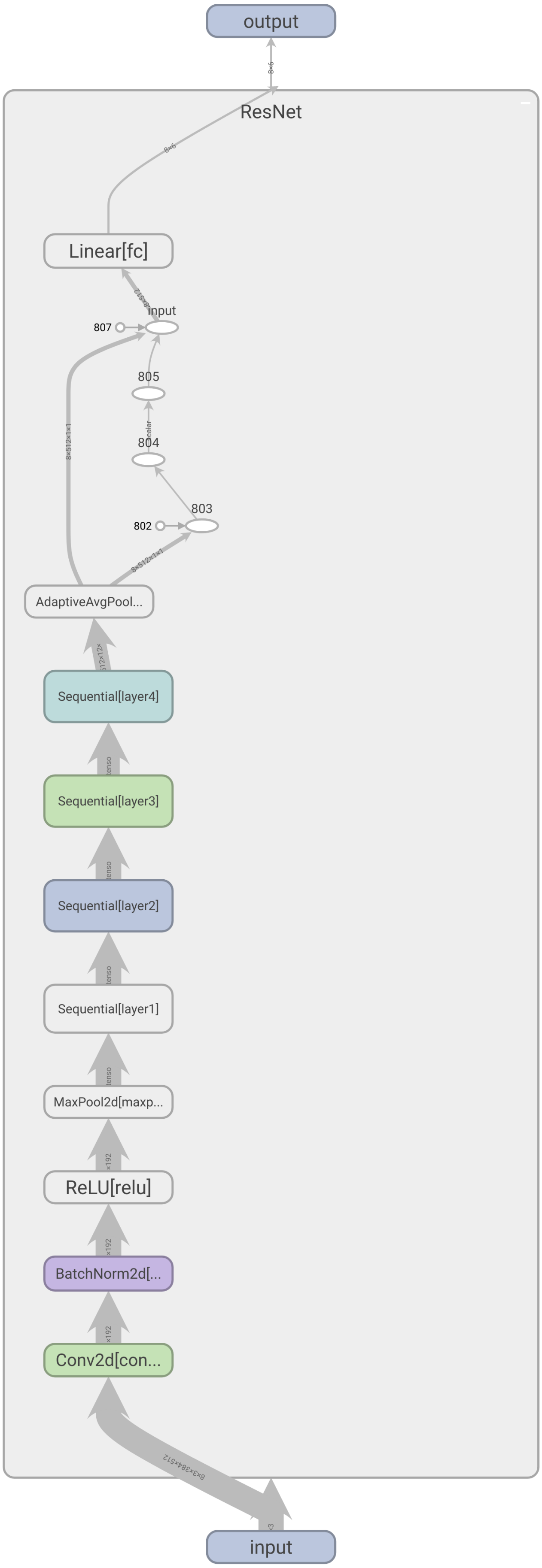


Fig. 6: ResNet18 Architecture

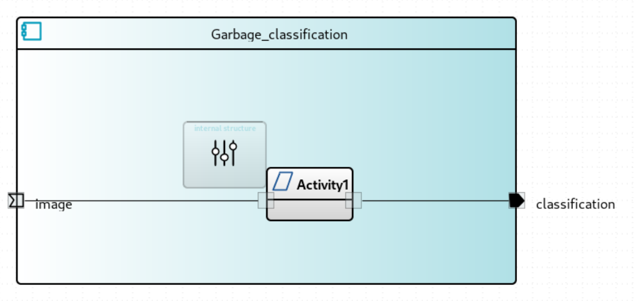


Fig. 7: ROS2 Papyrus Setup

for Python, thus, the model could be integrated into a pure C++ ROS2 envionment. Due to the flexibility of ROS2, the node was then able to perform classification on a live video stream from a camera or a prerecorded rosbag video stream.

The workflow of the ROS2 node was as follows. It takes an image as input, and then scales it to 512 x 384 to match the image sizes from TrashNet. The scaled image is then ran through the trained network. The ROS2 node then publishes a string with the classification result. The node can also be configured to show the input image with the classification result written onto it. The base ROS2 structure was created with Papyrus for Robotics. The activity setup can be seen in figure 7. The standalone C++ application follows a similar workflow, but can only take camera input, and writes the classification result to the terminal.

4 Experiments

The training results for the ResNet18 instance can be seen in figure 8 and the training results for the ResNet152 instance can be seen in figure 9. The memory usage of the networks can be seen in table 1.

For the ResNet18 instance, the learning rate was reduced from 0.001 to 0.0001 after the first epoch, and then from 0.0001 to 0.00001 after 2 more epochs, and then ran for 4 more epochs until it was clear the validation accuracy reached its asymptotic limit. The batch size was also decreased from 8 to 4 once the learning rate was decreased to 0.00001, to allow for finer grained network tuning. After the 2nd epoch, the accuracy began to converge around 80%, so all the layers were then unfrozen, and backpropagation was then applied to the entire network for the subsequent epochs. This can be seen by the drop in accuracy around the 500th epoch, followed by the steady increase in accuracy. A similar process was performed for the ResNet152 instance. The entire network was unfrozen after the 2nd epoch, and a drop in accuracy followed by a slow increase occurred. The batch size was also reduced late into the training, but it caused a slight drop in validation accuracy, so it was restored back to 8. The network showed much greater fluctuation in validation accuracy because it was much larger.

The ResNet 18 instance consumed a little more than half the memory that the ResNet152 instance consumed in inference and training of the last layer. The ResNet152 instance required an enourmous amount of memory to train after all layers were unfrozen, and a server with a more powerful GPU had to be used. The ResNet18 instance was trained on a gaming laptop with an Intel(R) Core(TM) i7-8750H CPU with 6 physical (12 logical) cores, 64GB RAM, and NVIDIA GeForce GTX 1070 GPU with 8GB of GPU memory. The ResNet152 instance was trained on a server with an Intel(R) Xeon(R) Gold 6254 CPU with 18 physical (36 logical) cores, 380GB RAM, and NVIDIA Tesla V100 GPU with 32GB of GPU memory.

The ROS2 node was then ran on the gaming laptop. The ResNet18 and ResNet152 instances were then ran both with and without GPU support. The inference speed of both models can be seen in table 2, and the validation accuracy on TrashNet can be seen in figure 3. The working ROS2 node can be seen in figure 10. While the ResNet152 network had a 3% higher validation accuracy on the TrashNet dataset than the RsNet18 instance did, it required roughly half a second to perform inference without a GPU. ResNet18 could achieve a real time performance of 15 FPS both with and without a GPU. The accuracy of the ResNet18 and 152 instances were then tested on prerecorded rosbag samples of garbage. The results for each video clip for ResNet18 and ResNet152 networks are shown in tables 4 and 6 respectively. The combined confusion matrices for the ResNet18 and ResNet152 networks on the video datasets are shown in figure 5 and 7 respectively. These numbers were obtained by replaying the rosbag while the classification node was running. Some messages were dropped, causing slight variance across different runs of the same data and allowed the ResNet18 to make more predictions than the ResNet152 network. However, the variance in the predictions was minor and did not affect the trends.

It can be seen that the ResNet152 network achieved a slightly higher total accuracy on the video dataset than the ResNet18 instance (85.45% vs 81.75%). For all classes except cardboard and trash, the ResNet18 network had high confusion with metal. The ResNet152 network had much better performance on glass than the ResNet18 network (99.42% vs 61.33%). The ResNet18 network performed especially poorly on the "Short Glass Jar 1" video clip. Both the "Short Glass Jar 1" and "Short Glass Jar 2" clips are of the same jar, but in the jar wrapper is a lot more visible in the "Short Glass Jar 1" clip. Thus, it possible that the network is confused by the wrappers on the containers. This could be mitigated by adding more training data, either with new pictures or by performing more data augmentation, or adding more advanced forms of regularization, like background cropping. Both ResNet18 and ResNet152 had good performance on most of the paper samples (88.22% and 88.30% accuracy respectively). ResNet18 had high confusion with metal on the "Crumpled Paper Receipt" video and ResNet152 had high confusion with cardboard on the "3 Crumpled Receipts" video. This once again could be due to limitations in the training data. ResNet18 achieved a 100% classification accuracy on the video cardboard samples while ResNet152 had some confusion with paper. It can be reasonable to assume that both cardboard and paper look similar, and that brown paper may look like cardboard. Despite this confusion, ResNet152's accuracy on cardboard is still quite high at 87.68%. ResNet18 demonstrated slightly better performance on plastic identification than ResNet152 (86.15% vs 77.48%). ResNet18 had consistent confusion with metal for the 2 uncrumpled plastic bottles but achieved 100% classification accuracy on the crumpled plastic bottle. Conversely, ResNet152 achieves >95% accuracy on both uncrumpled plastic bottles, but had high confusion with glass on the crumpled plastic bottle. While the TrashNet dataset has numerous images of plastic bottles, the majority of them are not crumpled. Thus, the network would benefit from more examples of plastic in different condition. Both networks had good performance on the video of metal (100% for ResNet18 and 97.12% for ResNet152). ResNet 152 had difficulty identifying the trash example, achieving an accuracy of only 8.33%. This could be because of the wide diversity of the image space of nonrecyclable trash. In addition, trash has the least number of samples in the TrashNet dataset, with only 137 images. Thus, it is reasonable to believe an increase in the number of trash training samples would improve network performance. It is interesting to note that ResNet18 exhibited good performance on the trash sample, with 99.53% accuracy. Another possible optimization to improve the classification of trash would be to check the logits of the other classes and predict trash if the logits of all the other classes are low.

5 Conclusions

In this paper, an application of CNNs to solve the problem of classifying recyclable garbage has been demonstrated. The network discussed herein demonstrates sim-

	ResNet18	**ResNet152**
Inference (C++)	828MiB	1018MiB
Inference (Python)	1415MiB	2771MiB
Training last layer	1553MiB	2717MiB
Training all layers	3457MiB	25127MiB

Table 1: Table of Model Memory Usage

	ResNet18	**ResNet152**
GPU	66.3408ms	68.0855ms
CPU	67.955ms	476.869ms

Table 2: Table of Execution Time

	ResNet18	**ResNet152**
TrashNet Validation Accuracy	92.52%	95.67%

Table 3: Table of Validation Accuracy

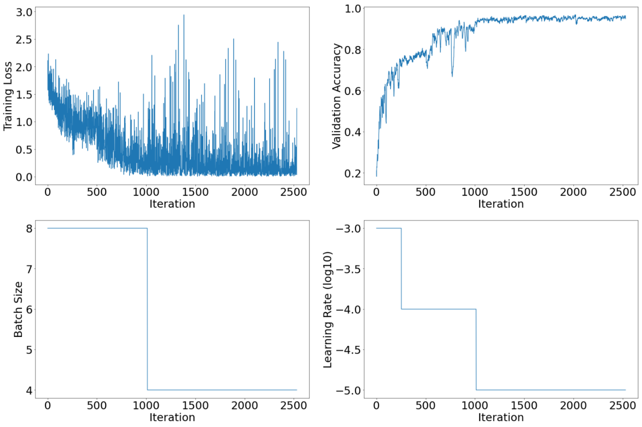


Fig. 8: ResNet18 Training

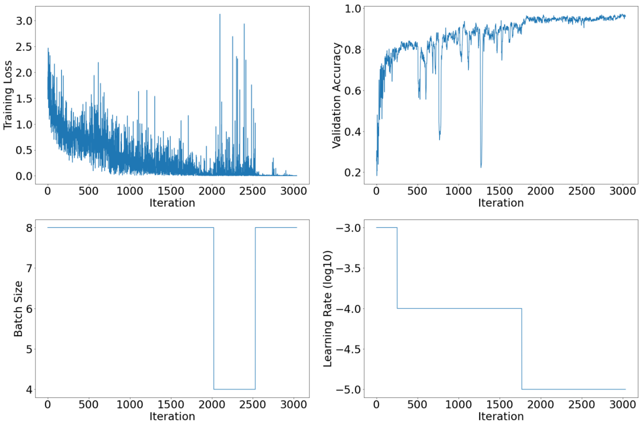
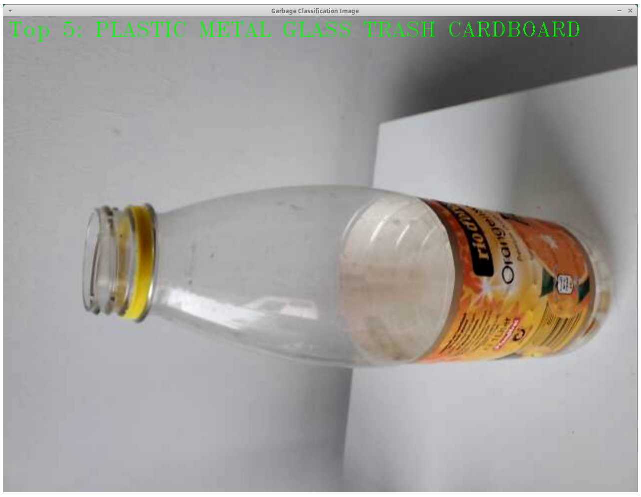


Fig. 9: ResNet152 Training



Top 5: PLASTIC METAL GLASS TRASH CARDBOARD

Fig. 10: ROS2 Node Output

ilar classification accuracy as previous approaches, and infrastructure to quickly train new architectures and apply trained architectures to classify garbage from camera feed has been presented. Using this infrastructure, trained networks that can generalize to unseen images of garbage are shown. All code and data involved in this project is publicly available[1].

This work exhibits the following limitations. The trained networks are still highly sensitive to the view angle, and slight changes in view angle can produce wrong predictions. Detection of unrecyclable trash can be unreliable, and the training data available for that category from TrashNet is also extremely limited. Current models also do not have any region of interest segmentation, and thus classification performance will be affected by how much image is background.

In the future, methods to quickly augment training data by pulling images out of rosbag video streams will be investigated. To shrink the space of unrecyclable garbage, the creation of a new compostable class and the separation of compostable materials from trash will be implemented into future training datasets. Additionally, to further enhance classification accuracy, the integration of foreground detection and cropping techniques will be performed. Finally, simplifications and faster architectures capable of performing on less computationally powerful embedded hardware will be explored.

[1] https://github.com/robmosys-tum/PapPercComp/tree/master/garbage_classification

	Predicted						Correct	Total	Accuracy
	Glass	Paper	Cardboard	Plastic	Metal	Trash			
Tall Glass Jar 1	205	0	0	0	69	0	205	274	74.82%
Short Glass Jar 1	8	0	0	0	366	0	8	374	2.14%
Dark Glass Bottle	286	0	0	0	103	0	286	389	73.52%
Tall Glass Jar 2	196	0	0	15	3	0	196	214	91.59%
Short Glass Jar 2	212	0	0	0	16	0	212	228	92.98%
Paper Magazine	0	268	0	0	20	0	268	288	93.06%
Paper Receipt	0	187	0	0	0	1	187	188	99.47%
Paper Newspaper	0	161	0	0	3	0	161	164	98.17%
Crumpled Newspaper	0	203	0	0	8	0	203	211	96.21%
Crumpled Paper Receipt	0	117	0	0	82	0	117	199	58.80%
3 Crumpled Receipts	0	232	0	0	42	0	232	274	84.67%
Small Cardboard Box	0	0	402	0	0	0	402	402	100.00%
Big Cardboard Box	0	0	179	0	0	0	179	179	100.00%
Plastic Juice Bottle	0	0	0	152	15	0	152	167	91.02%
Plastic Water Bottle	0	0	0	134	76	0	134	210	63.81%
Crumpled Plastic Bottle	0	0	0	280	0	0	280	280	100.00%
Metal Can	0	0	0	0	241	0	241	241	100.00%
Ice Cream Wrapper	0	0	0	0	1	210	210	211	99.53%

Table 4: Table of ResNet18 results on video dataset

	Predicted						Correct	Total	Accuracy
	Glass	Paper	Cardboard	Plastic	Metal	Trash			
Glass	907	0	0	15	557	0	907	1479	61.33%
Paper	0	1168	0	0	155	1	1168	1324	88.22%
Cardboard	0	0	581	0	0	0	581	581	100.00%
Plastic	0	0	0	566	91	0	566	657	86.15%
Metal	0	0	0	0	241	0	241	241	100.00%
Trash	0	0	0	0	1	210	210	211	99.53%
Total Accuracy							3673	4493	81.75%

Table 5: Confusion matrix of ResNet18 accuracy on video dataset

	Predicted						Correct	Total	Accuracy
	Glass	Paper	Cardboard	Plastic	Metal	Trash			
Tall Glass Jar 1	170	0	0	0	1	0	170	171	99.42%
Short Glass Jar 1	213	0	0	0	16	0	213	229	93.01%
Dark Glass Bottle	191	0	0	0	14	0	191	205	93.17%
Tall Glass Jar 2	98	0	0	0	18	0	98	116	84.48%
Short Glass Jar 2	116	0	0	0	0	0	116	116	100.00%
Paper Magazine	0	155	7	0	0	0	155	162	95.68%
Paper Receipt	0	109	2	0	0	0	109	111	98.20%
Paper Newspaper	0	82	10	0	0	0	82	92	89.13%
Crumpled Newspaper	0	120	0	0	0	0	120	120	100.00%
Crumpled Paper Receipt	0	112	0	0	0	0	112	112	100.00%
3 Crumpled Receipts	0	90	65	0	0	0	90	155	58.06%
Small Cardboard Box	0	43	196	0	0	0	196	239	82.00%
Big Cardboard Box	0	0	110	0	0	0	110	110	100.00%
Plastic Juice Bottle	0	0	4	104	0	0	104	108	96.30%
Plastic Water Bottle	0	0	5	106	0	0	106	111	95.50%
Crumpled Plastic Bottle	75	0	0	79	0	0	79	154	51.30%
Metal Can	0	4	0	0	135	0	135	139	97.12%
Ice Cream Wrapper	0	0	34	76	0	10	10	120	8.33%

Table 6: Table of ResNet152 results on video dataset

	Predicted						Correct	Total	Accuracy
	Glass	Paper	Cardboard	Plastic	Metal	Trash			
Glass	788	0	0	18	31	0	788	837	94.15%
Paper	0	668	84	0	0	0	668	752	88.30%
Cardboard	0	43	306	0	0	0	306	349	87.68%
Plastic	75	0	9	289	0	0	289	373	77.48%
Metal	0	4	0	0	135	0	135	139	97.12%
Trash	0	0	34	76	0	10	10	120	8.3%
Total Accuracy							2196	2570	85.45%

Table 7: Confusion matrix of ResNet152 accuracy on video dataset

References

1. Auto-trash sorts garbage automatically at the techcrunch disrupt hackathon. https://techcrunch.com/2016/09/13/auto-trash-sorts-garbage-automatically-at-the-techcrunch-disrupt-hackathon/ (2016)
2. The efficiency of densenet. https://medium.com/@smallfishbigsea/densenet-2b0889854a92 (2017)
3. Torchscript - pytorch documentation. https://pytorch.org/docs/stable/jit.html (2019)
4. Aral, R.A., Keskin, Ş.R., Kaya, M., Hacıömeroğlu, M.: Classification of trashnet dataset based on deep learning models. In: 2018 IEEE International Conference on Big Data (Big Data), pp. 2058–2062. IEEE (2018)
5. Bai, J., Lian, S., Liu, Z., Wang, K., Liu, D.: Deep learning based robot for automatically picking up garbage on the grass. IEEE Transactions on Consumer Electronics **64**(3), 382–389 (2018)
6. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251–1258 (2017)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR09 (2009)
8. 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, pp. 770–778 (2016)
9. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708 (2017)
10. 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, pp. 940–945 (2016)
11. Niu, G., Li, J., Guo, S., Pun, M.O., Hou, L., Yang, L.: Superdock: A deep learning-based automated floating trash monitoring system. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1035–1040. IEEE (2019)
12. Silva, A., Soares, E.: Artificial intelligence in automated sorting in trash recycling. XV Encontro Nacional de Inteligência Artificial e Computacional (2018)
13. Vo, A.H., Vo, M.T., Le, T., et al.: A novel framework for trash classification using deep transfer learning. IEEE Access **7**, 178631–178639 (2019)
14. Yang, M., Thung, G.: Classification of trash for recyclability status. CS229 Project Report **2016** (2016)