Machine learning and deep learning based domestic waste classification and target detection

Wei Huanzhe William
Data Science
BNU-HKBU United International College
n830026116@mail.uic.edu.cn

Tong Shengge Mclaren
Data Science
BNU-HKBU United International College
n830024236@mail.uic.edu.cn

***Abstract*—Domestic waste classification has been raised public concern due to the new policy from the government. People have to know the type of the garbage when they go to deal with the garbage. In this project, we plan to use both machine learning models and deep learning models to achieve the goals on detect the target of the specific garbage in the image and predict the type of the garbage to make the classification. In order to deal with the real-world problems, we attend the data open contest held by Zhuhai government and get the dataset. Our structure is divided in three parts. Firstly, is to preprocess the messy images. Secondly, we build a simple neural network based on Keras framework. Moreover, we build a deep learning models which is called CenterNet algorithm to do the target detection and using ResNet101 to do the classification to get a better performance compare to the simple BP neural network model.**

Index Terms – Multi-class Classification, Image preprocessing, Target detection, Neural network, ResNet101, CenterNet.

I. BACKGROUND AND MOTIVATION

Speaking about domestic waste classification. Most of the people think that it is too troublesome to identify all those kinds of garbage. However, government is already including the garbage classification in laws to help improve the quality of the environment. People should identify all the domestic waste when they want to throw them. In order to solve such problems, we consider to implement some machine learning models that we learned in class and some deep learning models to improve performance on classify the garbage.

Taking advantage of the recent open data contest held by Zhuhai government, we are able to attend the contest and get the data from the organization which is all from our daily lives. Since for the machine learning and deep learning models, the data are all images, we also do some data preprocessing before we put the data into the models.

In this project, we are focusing on the image classification. And due to the number of types of the garbage is about 40, actually, it is a multi-class classification. Firstly, we implement a simple Neural Network model and only classify the second level labels of the garbage. Then, we implement the deep learning algorithm ResNet101 【4】 to do the all-labels classification. The data preprocessing parts for these two models are divided separately. Besides the classification part, we also implement an advanced target detection deep learning algorithm which is called CenterNet.

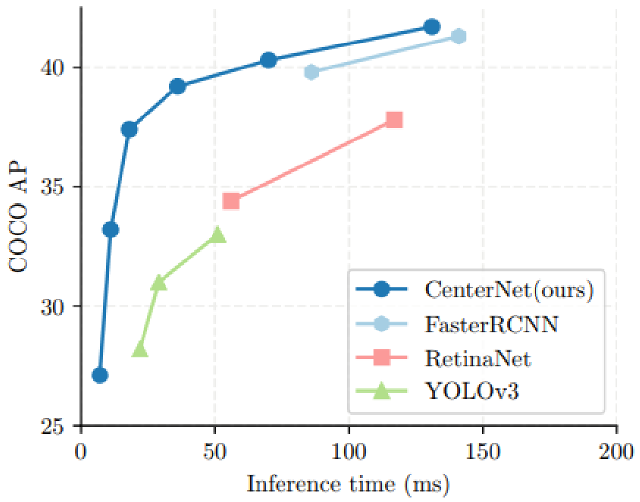


Fig. 1: Speed-accuracy trade-off on COCO validation for real-time detectors.

By comparing with the other target detection model, we found that CenterNet has a better performance. In the end, we also did some comparisons and future expectations. 【3】

II. RELATED WORK

The paper “Objects as Points” pointed that the model is built with the target as a point -- the central point of the target BBox. The detector uses key-point estimation to find the center point and returns to other target properties, such as size, 3D position, orientation, and even attitude. It uses the same model was used to make 3D BBox in Kitti Benchmark and human body posture detection in Coco Keypoint DataSet, compared with the complex multi-stage approach, our approach achieves competitive results in real time

The paper “Deep Residual Learning for Image Recognition” showed that the deeper the neural network, the harder it is to train. This paper proposes a residual learning framework to alleviate the difficulty of training deep networks (which are much deeper than previous networks).The researcher redefined the learning process of the layer as the learning of an input-dependent residual function rather than an input-independent function. Then they conducted comprehensive experiments to prove that this residual network is easier to optimize, and can achieve the accuracy improvement brought by depth (although the greater the depth, the higher the accuracy, but the experiment, the ordinary network is not the case). On the ImageNet dataset, the authors evaluated the residuals network (up to 152 layers, which is still very low in complexity despite being 8 times deeper than VGG).

III. Preliminary

A. *Simple Neural Network*

We get the dataset from Zhuhai Open Data innovation Apps Contest. It has the training set with about 5000 image data. And ten image data with three target garbage each for the test set. There are also a number of json files which contains the detailed information of each images including two level features and the size of each image. Here, in simple BP Neural Network model, we only classify the second level labels as the features of each of the image. We draw a bar chart to show the distribution of each features.

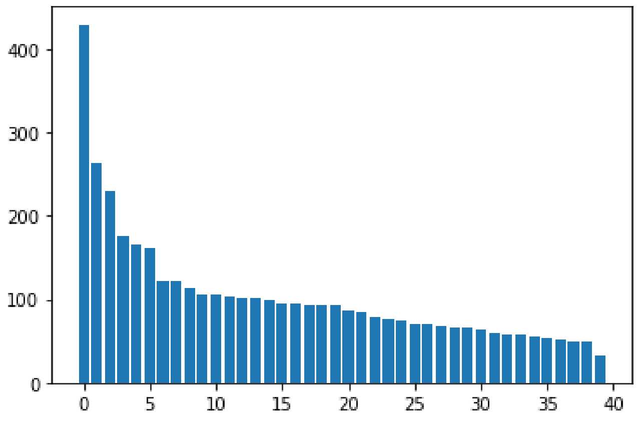


Fig. 2: Distribution of garbage

The x-Axis of the bar chart shown above represent the features of the images. And the y-Axis represents the quantity of image for each category. The number shown in the x-Axis represent in the following tables.

x-Axis of the bar chart	Features (categories/labels)
0	'塑料玩具'
1	'一次性快餐盒'
2	'毛绒玩具'
3	'塑料衣架'
4	'插头电线'
5	'烟蒂'
6	'竹筷'
…	…
34	'干电池'
35	'鱼骨'
36	'软膏'
37	'金属食品罐'
38	'调料瓶'
39	'过期药物'
40	'饮料瓶'

TABLE I: The categories of each garbage

There are totally forty categories for the whole dataset. We only show part of them in the above tables.

In the data preprocessing part, we also did some data cleaning and data normalization. There are some images actually have no labels, we just delete them from our dataset. Before the model implementation, we made all the images in the same size which is a kind of data normalization and we also turn the labels to the dummy variables which is actually a feature vector for each of the images. The feature vector combines with 0 and 1 and for every image, they are unique. After doing all the data preprocessing. We split the training set and validation set due to the 80/20 rule. Which is 80 percent of the whole dataset are used for model training and 20 percent of the whole dataset used to observe the prediction. The following images and vectors are a few examples that I choose from the dataset for have an overview.

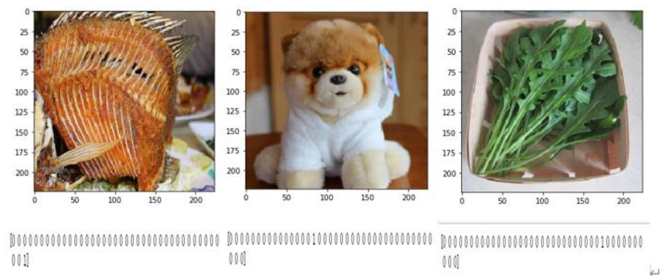


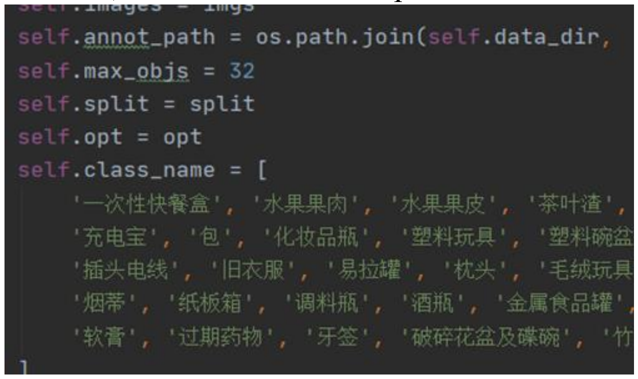
Fig. 3: Example training images and its corresponding feature vector

For the record, we did not turn the image into gray scale since we think that the color has a significant meaning for the images. And we resize every image to the size of 224 × 224 since according to our observation, the target is about in these sizes. For those of the images which is not three channels, we delete them from our dataset. The detailed of the data preprocessing part are shown in the codes. Feel free to check.

B. Deep learning

The data set is from Zhuhai Data Open and Innovative Application Competition, over 4000 pictures of domestic garbage. At first, we use 10 percent of the data as validation sets and the rest as training sets.

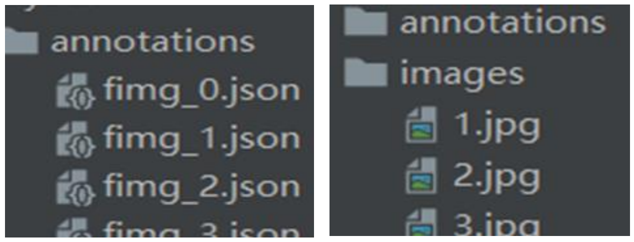
It has 40 classes and we use _init_ function to get the txt file, it stores the name of picture.



```
self.images = imgs
self.annot_path = os.path.join(self.data_dir,
self.max_objs = 32
self.split = split
self.opt = opt
self.class_name = [
    '一次性快餐盒', '水果果肉', '水果果皮', '茶叶渣',
    '充电宝', '包', '化妆品瓶', '塑料玩具', '塑料碗盆
    '插头电线', '旧衣服', '易拉罐', '枕头', '毛绒玩具
    '烟蒂', '纸板箱', '调料瓶', '酒瓶', '金属食品罐',
    '软膏', '过期药物', '牙签', '破碎花盆及碟碗', '竹
]
```

Fig. 4: Get txt files and 40 classes by _init_ function

We store json files and test pictures in file annotations and images respectively.

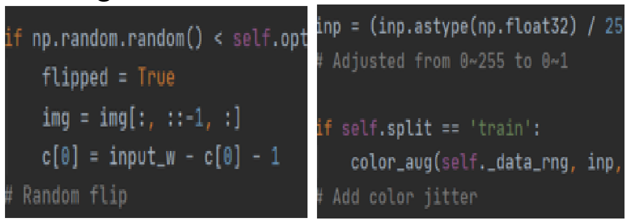


```
annotations
    fimg_0.json
    fimg_1.json
    fimg_2.json
    fimg_3.json
```

```
annotations
images
    1.jpg
    2.jpg
    3.jpg
```

Fig. 5, 6: json and jpg documents

Then we read the data by _getitem_ function. As the pictures they offered are too large to run, we resize the images into 128*128, padding is used when height and width are different.



```
if np.random.random() < self.opt
    flipped = True
    img = img[:, ::-1, :]
    c[0] = input_w - c[0] - 1
# Random flip
```

```
inp = (inp.astype(np.float32) / 25
# Adjusted from 0~255 to 0~1

if self.split == 'train':
    color_aug(self._data_rng, inp,
# Add color jitter
```

Fig. 7, 8: data enhancement

Then use methods such as scaling, flipping, center point migration, color dithering to enhance data.

IV. METHODOLOGY

A. Simple Neural Network

Neural network is actually a simple network which does not contains convolutional layers and pooling layers. It can be applied on the classification problems with vector or matrix as input. And for multi-class classification problems, it is easier than the SVM or logistic regression since they are used more in binary classification. We build this network based on the Keras framework with two fully connected layers as the hidden layers and without convolutional layers. 【2】Each layer contains 512 neurons for tuning the parameters.

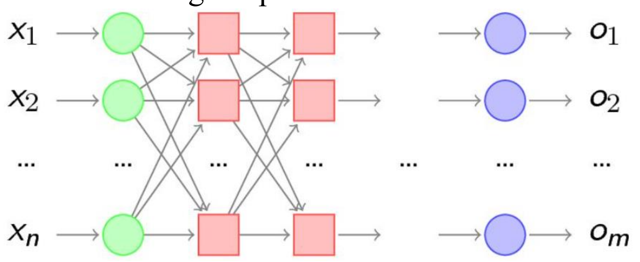
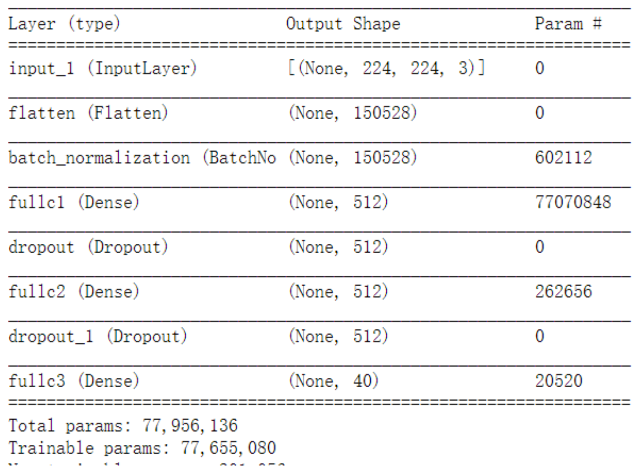


Fig. 9: Neural Network structure

The above figure shows a simple prototype of the simple neural network, the green circle represents the input layers, the red square represents the hidden layers and the blue circle is the output layers.

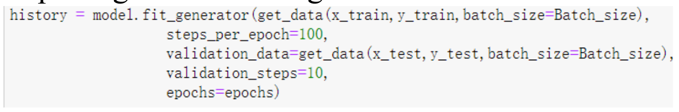


```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
flatten (Flatten)            (None, 150528)            0
_________________________________________________________________
batch_normalization (BatchNo (None, 150528)            602112
_________________________________________________________________
fullc1 (Dense)               (None, 512)               77070848
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
fullc2 (Dense)               (None, 512)               262656
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
fullc3 (Dense)               (None, 40)                20520
=================================================================
Total params: 77,956,136
Trainable params: 77,655,080
```

Fig. 10: Our construction on the network

The input the of for our simple neural network is the images that we unified the size of 224 × 224. And using flatten layers to flatten the data into one dimension. We also add the batch normalization layers and drop out layers alternate with the fully connected layers to avoid the overfitting problems. For the output, it is a feature vector represent the categories which length is 40. 【1】

In the training process, we use a generator instead of putting all the training data into the models.



```
history = model.fit_generator(get_data(x_train, y_train, batch_size=Batch_size),
                              steps_per_epoch=100,
                              validation_data=get_data(x_test, y_test, batch_size=Batch_size),
                              validation_steps=10,
                              epochs=epochs)
```

Fig. 11: Generator in training process

Here, get data is a function we defined for randomly pick the images from the training set. We can define the batch size by ourselves which is actually the number of images we randomly pick when doing one iteration in the training process. We define 100 iterations for each epoch. And We can try different batch size and epochs to make a comparison and use the best combination.

B. Deep learning

➢ *CenterNet Loss Function*

For each ground truth key point p ϵ $R^2$of class c, we compute a low-resolution $\tilde{p} = \left\lfloor \frac{p}{R} \right\rfloor$. 【4】Then splat all ground truth key points onto a heat map Y ∈ $[0,1]^{\frac{W}{R}*\frac{H}{R}*C}$ using a Gaussian kernel $Y_{xyc} = \exp(-\frac{(x-\tilde{p}_x)^2+(y-\tilde{p}_y)^2}{2\sigma_p^2})$, where $\sigma_p$is an object size-adaptive standard deviation. If two Gaussians of the same class overlap, we take the element-wise maximum. 【3】The training objective is a penalty-reduced pixel wise logistic regression with focal loss:

$$L_k = \frac{-1}{N}\sum_{xyc}\begin{cases}(1-\hat{Y}_{xyc})^\alpha \log(\hat{Y}_{xyc}) & \text{if } Y_{xyc}=1 \\ (1-Y_{xyc})^\beta (\hat{Y}_{xyc})^\alpha & \text{otherwise} \\ \log(1-\hat{Y}_{xyc}) & \end{cases} \quad (1)$$

Heat map loss is calculated using the idea of focal loss, where α and β are the hyper-parameters of focal loss, and where N is the number of positive samples for standardization.

All classes c share the same offset prediction. The offset is trained with an L1 loss in formula 2.

$$L_{off} = \frac{1}{N}\sum_{p}\left|\hat{O}_{\tilde{p}} - \left(\frac{p}{R} - \tilde{p}\right)\right|. \quad (2)$$

Let $(x_2^{(k)}, y_1^{(k)}, x_2^{(k)}, y_2^{(k)})$be the bounding box of object k with category cₖ. Its center point is lies at $p_k = (\frac{x_1^{(k)}+x_2^{(k)}}{2}, \frac{y_1^{(k)}+y_2^{(k)}}{2})$. Then use key point estimator $\hat{Y}$ to predict all center points. In addition, we regress to the object size $s_k = (x_2^{(k)} - x_1^{(k)}, y_2^{(k)} - y_1^{(k)})$ for each object k.

$$L_{size} = \frac{1}{N}\sum_{k=1}^{N}\left|\hat{S}_{p_k} - s_k\right|. \quad (3)$$

Formula 4 shows an overview of the network output:

$$L_{det} = L_k + \lambda_{size}L_{size} + \lambda_{off}L_{off}. \quad (4)$$

➢ *Bounding boxes*

Each target is described by a central key point and a diagonal point.

The detected central key points are used to rearrange Top-k Bboxes to filter out incorrect Bboxes. The process is as follows:

1) Select Top-k central key points according to the score;
2) According to the corresponding offset value, these selected central key points are redrawn on the input image;
3) Design a central area for each Bbox, and check whether this central area contains the central key

points. It should be noted that the category of the central key points tested should be consistent with the category of Bbox;

4) If the center key-point is detected in the center area, the Bbox will be retained. The score value of this Bbox is equal to the average score of three points (top-left, bottom-right, center key-point). If no central key point is detected in the central area, the Bbox will be removed.

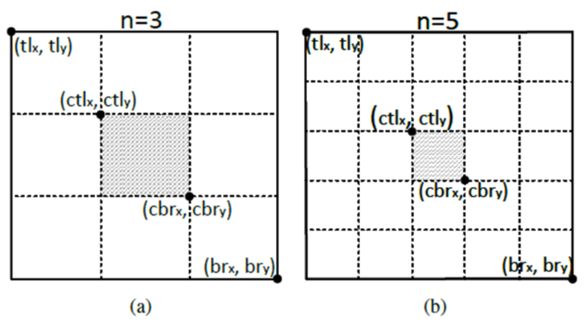
The size of the central region of Bbox will affect the detection results. For example, a small central region will lead to the ground recall of a small Bbox. Taking a large central region of a large Bbox will lead to low precision. We proposes that the scale-aware (adjustable size) central area comes from the Bbox which can adapt to various sizes. That is, for a small BBox box with a relatively large central area, for a large BBox box with a relatively small central area. 【5】

The formula used is as follows, involving the upper left corner and the lower right corner of the prediction box.

The coordinates of the upper left corner and the lower right corner of the central region:

$$\begin{cases} \mathrm{ctl_x} = \dfrac{(n+1)\mathrm{tl_x} + (n-1)\mathrm{br_x}}{2n} \\ \mathrm{ctl_y} = \dfrac{(n+1)\mathrm{tl_y} + (n-1)\mathrm{br_y}}{2n} \\ \mathrm{cbr_x} = \dfrac{(n-1)\mathrm{tl_x} + (n+1)\mathrm{br_x}}{2n} \\ \mathrm{cbr_y} = \dfrac{(n-1)\mathrm{tl_y} + (n+1)\mathrm{br_y}}{2n} \end{cases} \tag{1}$$

Here n is odd; When bbox is less than 150, n=3; And when bbox is greater than 150, n=5;



Calculate the central area according to formula 1, which is used to detect whether the central key point is included.

V. EXPERIMENTAL STUDY AND RESULT ANALYSIS

A. Simple Neural Network

From many neural network's experimental analysis, people usually first analyze the validation accuracy during the training process. And then observe the accuracy on the test set. After tuning the parameter to achieve a better accuracy, we start to predict the categories in the test set. We followed this procedure when doing the result analysis.

First, we tried batch size 4 and epochs 20, but it doesn't converge good. When we tired batch size 256 and epochs 50, the kernel will die. So, it means that we should choose a proper batch size and epochs to reach a relatively good result. Finally, we decided to use batch size 128 and epochs 50. The results are shown below. 【3】 We can see that the validation accuracy converges much slower than the training accuracy, so as the loss. We think this is not only because of the data inconsistency, but also because the simple neural network has limitation on the layers' adjustment. It cannot have a better convergence due to the simple structure of the network.

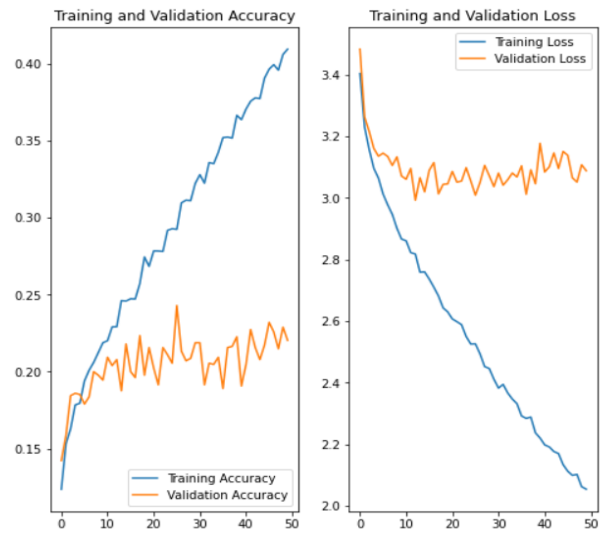


Fig. 12: Training and validation accuracy/loss

There are some of the test sets for us to predict the categories of the garbage. There are ten images in the test set and each image has three garbage targets.

Fig. 12: Test set image

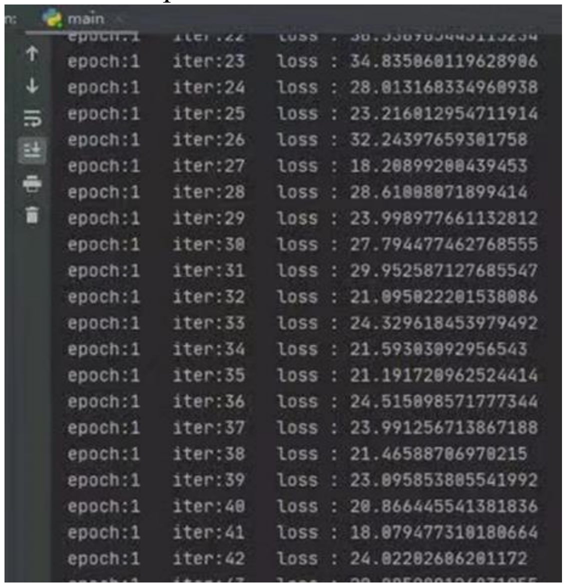
We do some image cropping for the test images and cut three targets as three images. We resize the images to 224 × 224 and put them into one array to do the model prediction. Finally, we get the following results.

```
Picture1.png's three categories are':塑料玩具,污损塑料,污损塑料
Picture2.png's three categories are':化妆品瓶,塑料衣架,枕头
Picture3.png's three categories are':塑料玩具,充电宝,枕头
Picture4.png's three categories are':包,洗发水瓶,塑料玩具
Picture5.png's three categories are':快递纸袋,包,插头电线
Picture6.png's three categories are':食用油桶,易拉罐,插头电线
Picture7.png's three categories are':锅,洗发水瓶,污损塑料
Picture8.png's three categories are':一次性快餐盒,食用油桶,塑料衣架
Picture9.png's three categories are':一次性快餐盒,塑料碗盆,一次性快餐盒
Picture10.png's three categories are':快递纸袋,一次性快餐盒,食用油桶
```

Fig. 13: Prediction for the test images

B. Deep learning

We create the model and train it. The lr is 1.25e-4, it decreases at the 45th and 60th 60epochs. The total training epoch is 70, the batch_size is 32, and we use Adam as the optimizer.

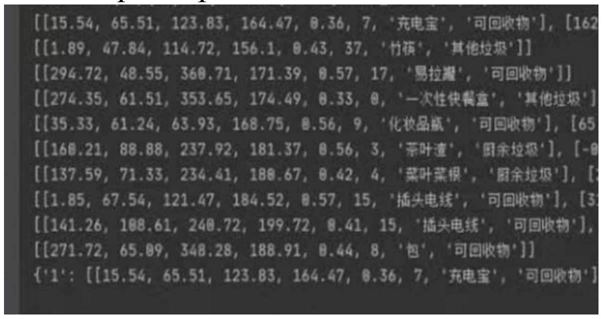


```
epoch:1    iter:23    loss : 34.835060119628906
epoch:1    iter:24    loss : 28.013168334960938
epoch:1    iter:25    loss : 23.216012954711914
epoch:1    iter:26    loss : 32.24397659301758
epoch:1    iter:27    loss : 18.20899200439453
epoch:1    iter:28    loss : 28.61008071899414
epoch:1    iter:29    loss : 23.998977661132812
epoch:1    iter:30    loss : 27.794477462768555
epoch:1    iter:31    loss : 29.952587127685547
epoch:1    iter:32    loss : 21.095022201538086
epoch:1    iter:33    loss : 24.329618453979492
epoch:1    iter:34    loss : 21.59303092956543
epoch:1    iter:35    loss : 21.191720962524414
epoch:1    iter:36    loss : 24.515098571777344
epoch:1    iter:37    loss : 23.991256713867188
epoch:1    iter:38    loss : 21.46588706970215
epoch:1    iter:39    loss : 23.095853805541992
epoch:1    iter:40    loss : 20.866445541381836
epoch:1    iter:41    loss : 18.079477310180664
epoch:1    iter:42    loss : 24.02202606201172
```

Fig. 14: Loss values of the model

It cost a long time to do the training as ResNet101 has so many layers. We get 40 heat-maps and WH of corresponding categories through the network.

The transform the obtained feature map decode into the required prediction box.



```
[[15.54, 65.51, 123.83, 164.47, 0.36, 7, '充电宝', '可回收物'], [162
[[1.89, 47.84, 114.72, 156.1, 0.43, 37, '竹筷', '其他垃圾']]
[[294.72, 48.55, 360.71, 171.39, 0.57, 17, '易拉罐', '可回收物']]
[[274.35, 61.51, 353.65, 174.49, 0.33, 0, '一次性快餐盒', '其他垃圾']
[[35.33, 61.24, 63.93, 168.75, 0.56, 9, '化妆品瓶', '可回收物'], [65
[[160.21, 88.88, 237.92, 181.37, 0.56, 3, '茶叶渣', '厨余垃圾'], [-0
[[137.59, 71.33, 234.41, 180.67, 0.42, 4, '菜叶菜根', '厨余垃圾'], [
[[1.85, 67.54, 121.47, 184.52, 0.57, 15, '插头电线', '可回收物'], [3
[[141.26, 108.61, 240.72, 199.72, 0.41, 15, '插头电线', '可回收物'],
[[271.72, 65.09, 348.28, 188.91, 0.44, 8, '包', '可回收物']]
{'1': [[15.54, 65.51, 123.83, 164.47, 0.36, 7, '充电宝', '可回收物']
```

Fig. 15: Output in list

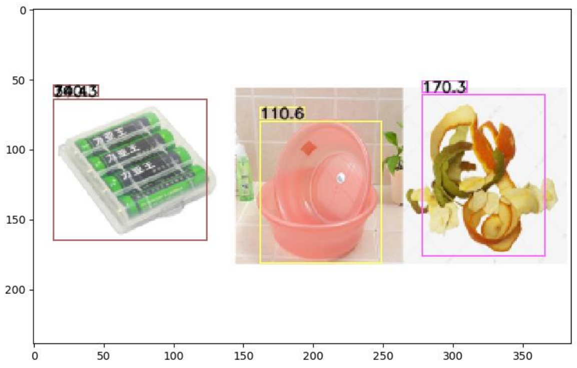


Fig. 16: Output1

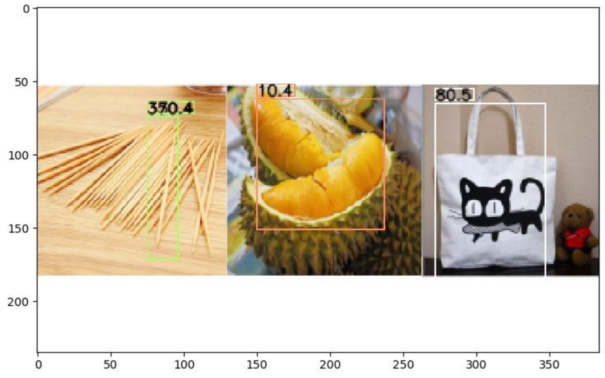


Fig. 17: Output2

After that, we screen the prediction box, excluding the prediction box with low confidence, and finally obtaining the category and coordinates of the prediction box above. In the left corner of the picture, the number means "category + confidence coefficient", for example, 110.6 means the category number is 11 and the confidence coefficient is 0.6

The accuracy of the test set is about 60%, and the validation set is about 80%. The main reason is that the training set is so small and the garbage pictures are not standard.

VI. CONCLUSION AND FUTURE WORK

For our whole project, we actually have 4 main parts, Data cleaning and preprocessing, simple neural network, ResNet101 and CenterNet target detection. And simple neural network performs bad on the image classification. Deep learning ResNet101 has lots of convolutional layers, it helps to tune the parameters during the training in a more efficient way. However, each of the parts has some difficulties, we have some future expectations and works to do for overcome these difficulties. For simple neural network, due to the memory overflow, we did not see the final convergence for the training accuracy, so maybe in the future, we can find some advanced computing resources and see do the final results. Even more, we can design more layers to compare the results. For Target detection, since CenterNet is already an advanced algorithm, we may try more models in the future and make some comparison to choose the best one. Last but not least, is the data collection parts. The data the committee of the contest gave us has some inconsistency problems; plastic toys have a significantly more data than the other categories. And the whole dataset is not enough since it has only about 5000 image data. And as you can see, our daily lives data is very hard to categorize. It is very different from the facial recognition, car plate recognition or traffic sign classification. For instance, for car plate, the pixel of 123 or ABC is very easy and similar to identify for the models. However, for our domestic trash, the category of toys can be verified in many kinds of toys, the matrix transform from the pixel is complex and different from each other. So, it is very difficult to achieve an accurate result. For the future work for this part, we have to enlarge the dataset for each of the categories and avoid the inconsistency problems, and more importantly, we have to try to make the images looks similar to avoid the noises from the environment.

REFERENCES

1. Bobulski J., Kubanek M. (2019) Waste Classification System Using Image Processing and Convolutional Neural Networks. In: Rojas I., Joya G., Catala A. (eds) Advances in Computational Intelligence. IWANN 2019. Lecture Notes in Computer Science, vol 11507. Springer, Cham. https://doi.org/10.1007/978-3-030-20518-8_30
2. B. Gan and C. Zhang, "Research on the algorithm of urban waste classification and recycling based on deep learning technology," 2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL), 2020, pp. 232-236, doi: 10.1109/CVIDL51233.2020.00-95.
3. Kaiwen DuanBai, Lingxi Xie, Honggang Qi, Qingming Huang, Qi TianSong. (2019). CenterNet: Keypoint Triplets for Object Detection. https://arxiv.org/abs/1904.08189
4. Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. (2015). Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385
5. Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, Philip Torr. (2019). Res2Net: A New Multi-scale Backbone Architecture. https://arxiv.org/abs/1904.01169