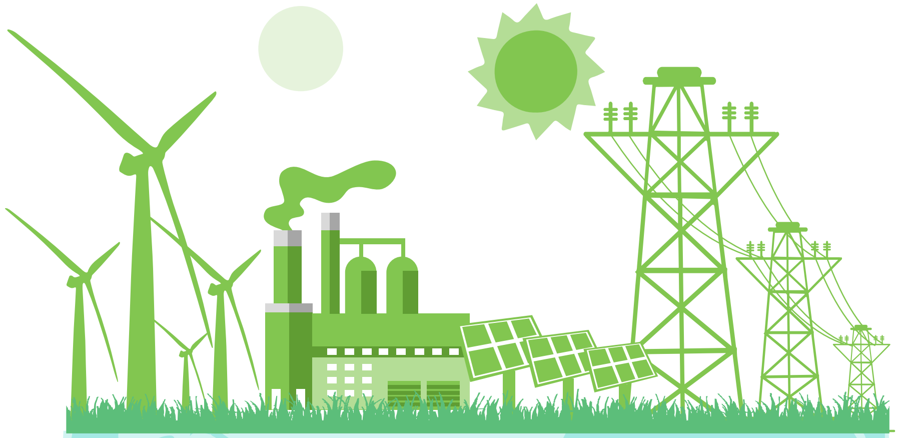


資源回收分類辨識

109034541
Group 9
蘇詠心

Agenda

01 研究動機與目的

5W1H

02 資料前處理

資料集介紹 / 資料視覺化 / 資料標準化 / 資料增強 / 資料切割

03 模型建立

CNN Model

04 超參數優化

Optimizer / batch size / number of layer / dropout rate

05 結論

模型結果 / 未來展望 / 未來應用

研究動機與目的

研究背景

根據統計，全球每年平均製造出21億噸垃圾，
足以填滿82萬座奧林匹克規格的游泳池

5W1H

WHY
由於目前資源回收率甚低，透過機器協助垃圾分類辨識，以提升民眾資源回收的意願。另可協主資源回收處裡業主在垃圾分類上具有更高的正確度與效率

WHERE
- 垃圾桶
- 資源回收場

WHO
一般民眾
資源回收處理業主

WHEN
進行垃圾分類前

WHAT
協助其進行垃圾分類

HOW
運用Python與CNN架構建立資源回收種類辨識模型

資料前處理

資料集介紹

Garbage Classification

- 使用 Kaggle 公開資料集
- 將資源回收類型分為 5 類
- Cardboard, Glass, Metal, Paper, Plastic
- 共 2000 筆資料

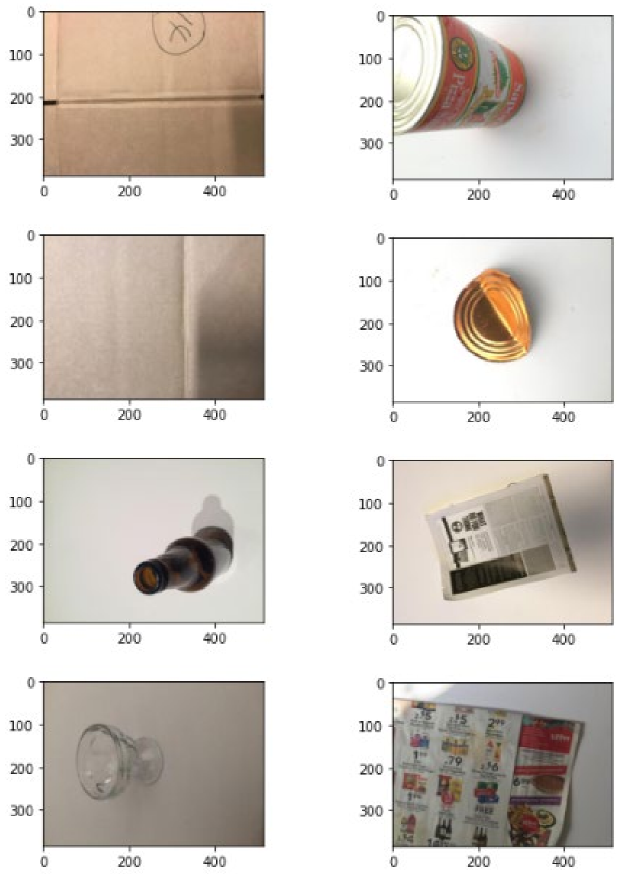
研究架構


資料前處理
資料視覺化
Encoding
資料標準化
資料增強
資料分割
CNN 模型建構
超參數調整
模型評估
模型建立

資料視覺化

使用Matplotlib 將圖片顯示出來

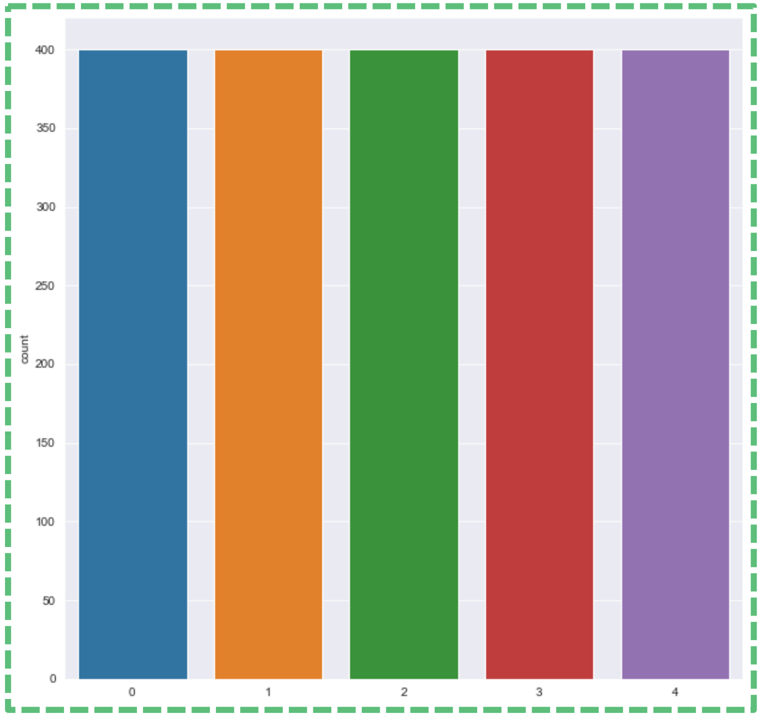
```python
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
folder = os.listdir('C:/Users/fr318/data_garbage_01/Garbage classification/Garbage classification')

for i in range(5):
    for j in range(2):
        j+=1
        plt.figure(figsize = (3, 3))
        folder = os.path.join(str(dir_path+'/'+labels[i]+'/'+labels[i]+str(j)+'.jpg'))
        img = mpimg.imread(folder)
        imgplot = plt.imshow(img)
        plt.show()
```



資料視覺化

- 以seaborn套件成現資料分布情形
- 確認無資料不平衡問題

```python
import seaborn as sns
plt.figure(figsize=(10,10))
sns.set_style("darkgrid")
sns.countplot(x=data)
```



資料標準化

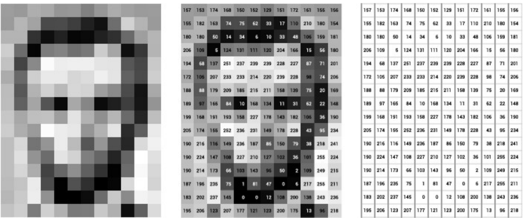
把像素值放縮到 0 和 1 之間有利於模型的收斂

```
train=ImageDataGenerator(horizontal_flip=True,
                         vertical_flip=True,
                         validation_split=0.1,
                         rescale=1./255,
                         shear_range = 0.1,
                         zoom_range = 0.1,
                         width_shift_range = 0.1,
                         height_shift_range = 0.1,)

test=ImageDataGenerator(rescale=1/255,
                        validation_split=0.1)
```

色碼各個像素值範圍介於 0~255，

故將所有特徵值除上 255 ，可縮減

數據間的跨度



資料增強

利用 ImageDataGenerator 生成更多圖像

```
#Data Augmentation
train=ImageDataGenerator(horizontal_flip=True,
                         vertical_flip=True,
                         validation_split=0.1,
                         rescale=1./255,
                         shear_range = 0.1,
                         zoom_range = 0.1,
                         width_shift_range = 0.1,
                         height_shift_range = 0.1,)

test=ImageDataGenerator( validation_split=0.1,
                         rescale=1/255)


train_generator=train.flow_from_directory(dir_path,
                                          target_size=(300,300),
                                          batch_size=32,
                                          class_mode='categorical',
                                          subset='training')

test_generator=test.flow_from_directory(dir_path,
                                        target_size=(300,300),
                                        batch_size=32,
                                        class_mode='categorical',
                                        subset='validation')
```

- 隨機對圖片執行水平/垂直翻轉操作
- 把像素值放縮到0和1之間有利於模型的收斂
- 錯切變換
- 讓圖片在長或寬的方向進行放大
- 水平位置平移和上下位置平移

資料切割

將資料分為 training set 和 testing set

90%

1,800 images

Training Set

10%

200 images

Testing Set

模型建立

模型建立

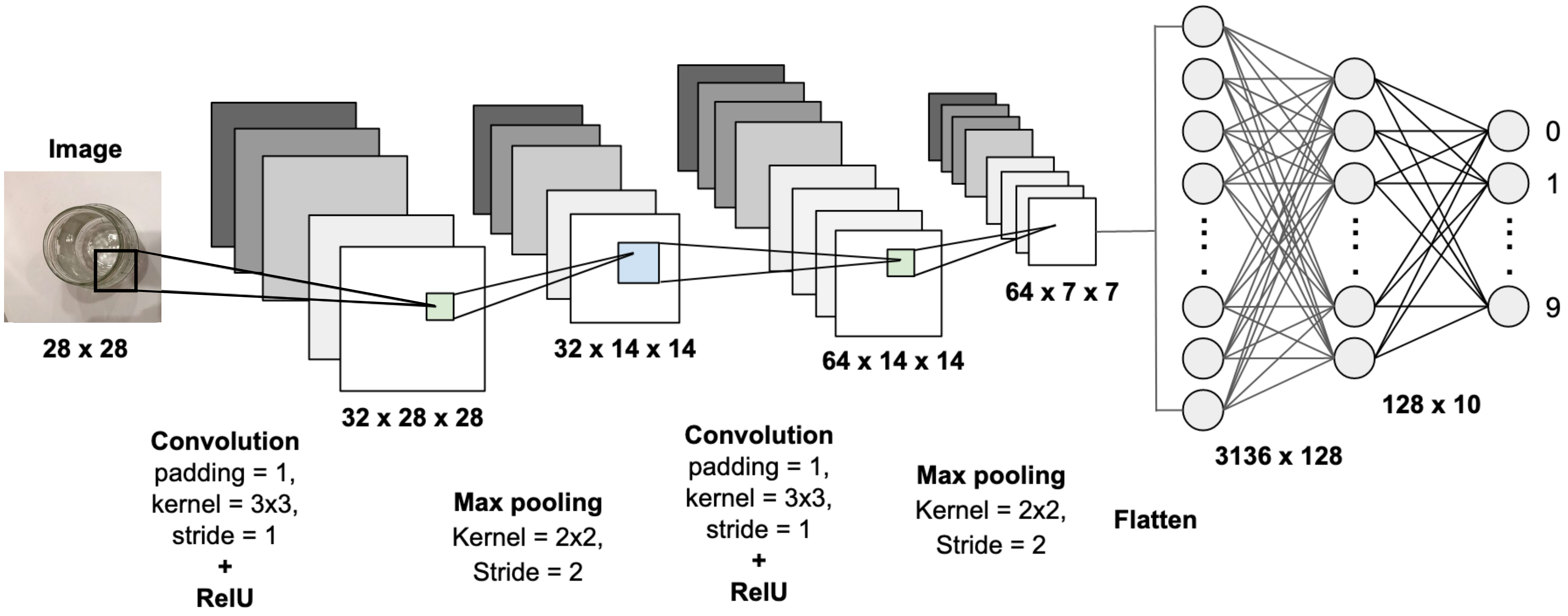


Image
28 x 28
32 x 28 x 28
Convolution
padding = 1,
kernel = 3x3,
stride = 1
+
RelU
32 x 14 x 14
Max pooling
Kernel = 2x2,
Stride = 2
64 x 14 x 14
Convolution
padding = 1,
kernel = 3x3,
stride = 1
+
RelU
64 x 7 x 7
Max pooling
Kernel = 2x2,
Stride = 2
Flatten
3136 x 128
128 x 10
0
1
9

模型建立

```
model=Sequential()
#Convolution blocks

model.add(Conv2D(32,(3,3), padding='same',input_shape=(300,300,3),activation='relu'))
model.add(MaxPooling2D(pool_size=2))
#model.add(SpatialDropout2D(0.5)) # No accuracy

model.add(Conv2D(64,(3,3), padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))

model.add(Conv2D(32,(3,3), padding='same',activation='relu'))
model.add(MaxPooling2D(pool_size=2))

#Classification layers
model.add(Flatten())

model.add(Dense(64,activation='relu'))
#model.add(SpatialDropout2D(0.5))
model.add(Dropout(0.2))

model.add(Dense(32,activation='relu'))
model.add(Dropout(0.2))

model.add(Dense(5,activation='softmax'))

filepath="trained_model22.h5"
checkpoint1 = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, save_best_only=True, mode='max')
callbacks_list = [checkpoint1]

model.summary()
```

- Convolution (# of neuron=32)
- Max-pooling
- Convolution (# of neuron=64)
- Max-pooling
- Convolution (# of neuron=32)
- Max-pooling
- Flatten
- Dense (# of neuron=64)
- Dropout = 0.2
- Dense (# of neuron=32)
- Dropout = 0.2
- Output Layer

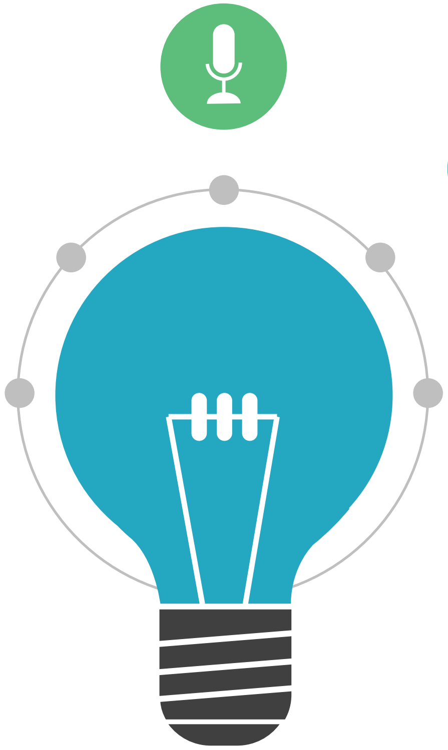
超參數優化

超參數優化

Dropout Rate
0.01 / 0.1 / 0.2 / 0.3 / 0.5

Activation Function
ReLU / Softmax

Optimizer
Adam / adagrad



Loss Function
categorical_crossentropy

Batch Size
32 / 50 / 100

超參數調整

調整 optimizer，發現使用 adam 結果較好

Convolution	Max-pooling	Dense	droupout rate	activation function	loss function	optimizer	metrics	epochs	batch size	accuracy
3	3	3	0.2	relu	categorical_ crossentropy	adam	acc	100	32	76.50%
3	3	3	0.2	relu	categorical_ crossentropy	adagrad	acc	100	32	49%

超參數調整

調整 batch size，當 batch size 為 32 時結果較佳

Convolution	Max-pooling	Dense	droupout rate	activation function	loss function	optimizer	metrics	epochs	batch size	accuracy
3	3	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	76.50%
3	3	3	0.2	relu	categorical_crossentropy	adam	acc	100	50	69.50%
3	3	3	0.2	relu	categorical_crossentropy	adam	acc	100	100	74%

超參數調整

嘗試使用不同層數之convolution、max-pooling 和 dense

Convolution	Max-pooling	Dense	droupout rate	activation function	loss function	optimizer	metrics	epochs	batch size	accuracy
2	2	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	71%
3	3	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	76.50%
4	4	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	80%
5	5	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	73%
5	5	4	0.2	relu	categorical_crossentropy	adam	acc	100	32	77%

超參數調整

調整 dropout，當 dropout 為 0.2 時結果較佳

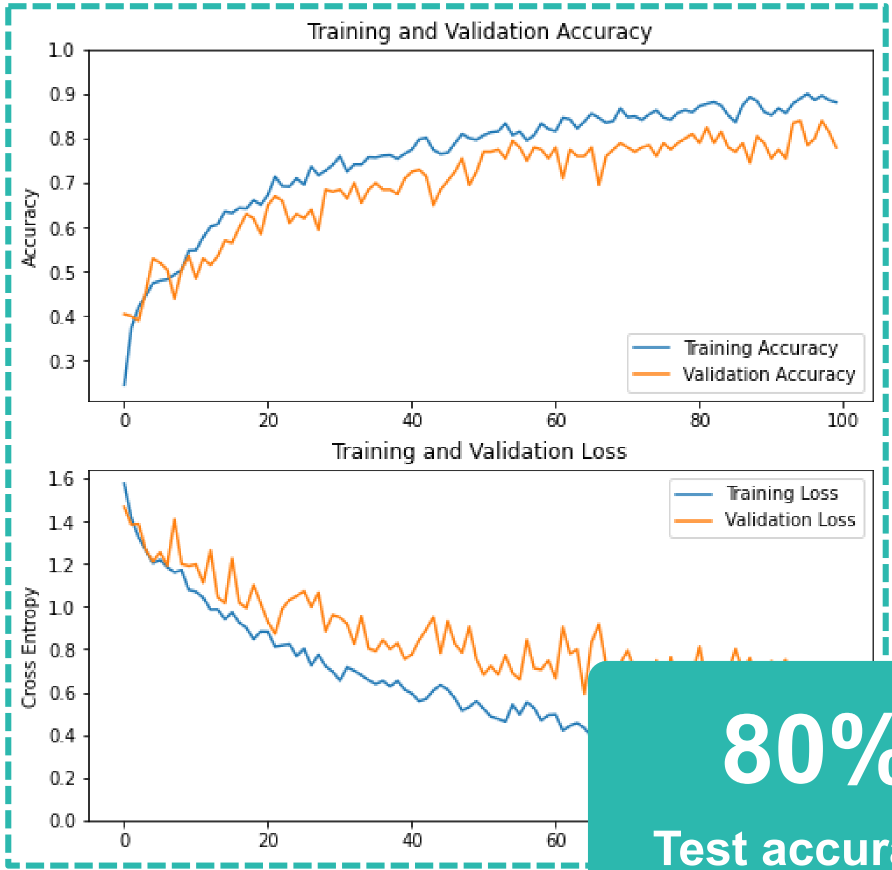
Convolution	Max-pooling	Dense	droupout rate	activation function	loss function	optimizer	metrics	epochs	batch size	accuracy
4	4	3	0.01	relu	categorical_crossentropy	adam	acc	100	32	74%
4	4	3	0.1	relu	categorical_crossentropy	adam	acc	100	32	78.50%
4	4	3	0.2	relu	categorical_crossentropy	adam	acc	100	32	80%
4	4	3	0.3	relu	categorical_crossentropy	adam	acc	100	32	77.50%
4	4	3	0.5	relu	categorical_crossentropy	adam	acc	100	32	68.50%

結論

超參數優化結果

將 test accuracy 與 loss 結果繪製出來

```
#Accuracy graph
acc = history.history['acc']
val_acc = history.history['val_acc']

loss = history.history['loss']
val_loss = history.history['val_loss']

# ________________ Graph 1 ________________

plt.figure(figsize=(8, 8))
plt.subplot(2, 1, 1)
plt.plot(acc, label='Training Accuracy')
plt.plot(val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.ylabel('Accuracy')
plt.ylim([min(plt.ylim()),1])
plt.title('Training and Validation Accuracy')

# ________________ Graph 2 ________________

plt.subplot(2, 1, 2)
plt.plot(loss, label='Training Loss')
plt.plot(val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.ylabel('Cross Entropy')
plt.ylim([0,max(plt.ylim())])
plt.title('Training and Validation Loss')
plt.show()
```



80%

Test accuracy

未來展望

增加資料集數量

增加資料集數量，以提升模型訓練之準確度

超參數優化

可利用貝葉斯超參數優化方法，找出更適合此模型之超參數

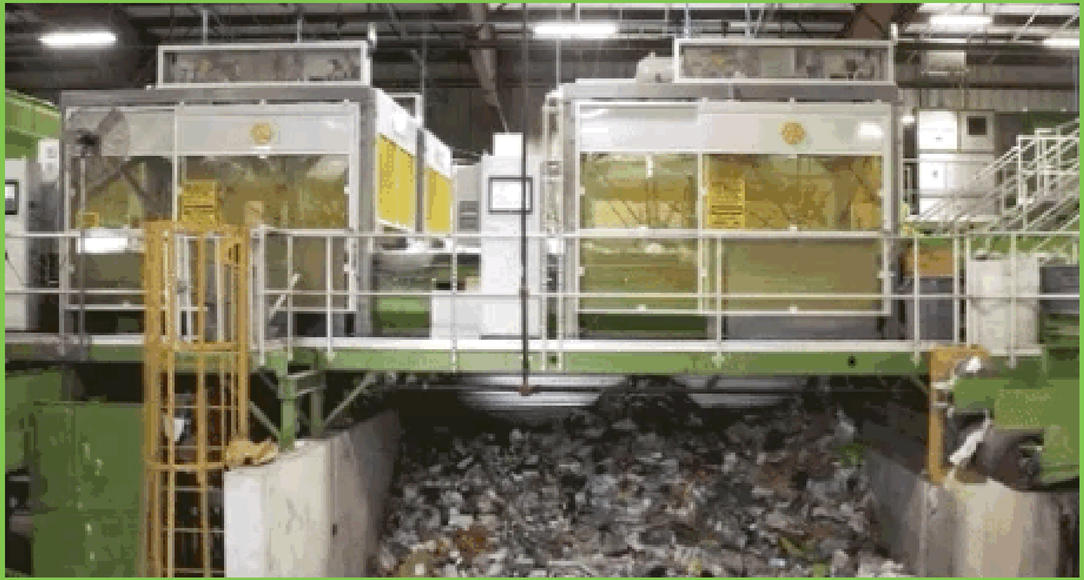
提升模型準確度

未來應用

將資源回收種類辨識系統應用至實際生活，
協助進行垃圾分類，並間接提高人們回收意願

資源回收
分類垃圾桶

垃圾場分類



Thank You