MAIS 202 – Project Deliverable 2

Problem Statement

The project is classifying the garbage/recycling according to its category (carboard, glass, metal, paper, plastic, and trash). I will classify them using CNN (Convolutional Neural Network). And before feeding the data into CNN, I will do image preprocessing to rescale the image and to turn image into black and white pictures.

Data Preprocessing

Data set to be used is Kaggle Garbage Classification data set (https://www.kaggle.com/asdasdasasdas/garbage-classification). Initially, wanted to work with this dataset and maybe add more data, for example, adding another classification, Styrofoam. However, getting hundreds of images for Styrofoam that will work with my data set would be a problem, unless I find dataset that is pre-made online. Therefore, I will stick to Kaggle's data set, which will be enough on its own. Kaggle's Garbage Classification dataset contains 6 classifications: carboard(393), glass (491), metal (400), paper(584), plastic(472), and trash (127), with all classifications having more than hundred data.

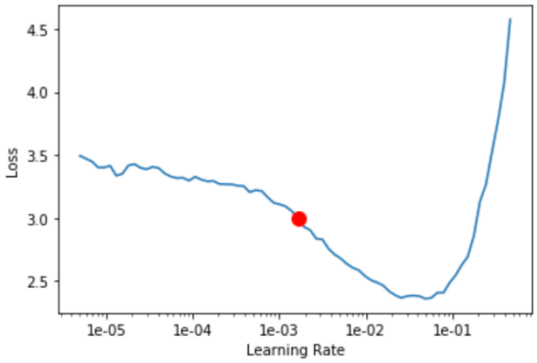
Data preprocessing will be done with a few steps: image resizing, gray-scaling, data augmentation, data segmentation. First, image resizing will be done with openCV library. Next, noise removing will be done with Gaussian blur/Gaussian smoothing technique in order to enhance image structures at different scales.

Machine learning model

CNN will be used as proposed in deliverable 1. There are some pre-trained models which could be used. Resnet34(https://www.kaggle.com/pytorch/resnet34) is one of the option. "A residual neural network is a convolution neural network with lots of layers. In particular, resnet34 is a CNN with 34 layers that's been pretrained on the ImageNet database. " – (Collin Ching, 2019). A pretrained CNN could be very powerful. The idea of Resnet is to adjust weights and bias in suboptimal values for nodes in a layer, which adjustments are only made to nodes if they need it (when there's non-zero residuals). When adjustment is needed, the residual function will operate to produce the residual of the operation to correct the prediction to match the real one.

The training/validation/test split was 50/25/25 and I randomly sorted the pictures into 3 different folders.

With **lr_find**, the optimal learning rate was found.



Then, **fit_one_cycle** was used to train and reduce the error_rate. It was ran for 3 epochs. Running for more epochs will reduce the error_rate but since one epoch took a lot of time, I did not have time for run like 20 epochs.

epoch	train_loss	valid_loss	error_rate	time
0	1.717028	0.634945	0.233333	30:19
1	1.092176	0.574047	0.182540	30:13
2	0.799799	0.491579	0.155556	30:19

I believe my model is a bit underfitting.

Preliminary results
Confusion matrix was used to get the accuracy of the model on test set.

```
conf_mat = confusion_matrix(y,y_pred)
print(conf_mat)
```

```
[[ 85   1   1   8   0   6]
 [  0  89  25   1  10   1]
 [  2  13  81   3   0   4]
 [  1   1   4 140   0   3]
 [  0  18   4   3  94   2]
 [  3   1   2   6   1  22]]
```

And the accuracy was about 80% which is not very good.

```
correct = 0

for r in range(len(conf_mat)):
  for c in range(len(conf_mat)):
    if (r==c):
      correct += conf_mat[r,c]

accuracy = correct/sum(sum(conf_mat))
print(accuracy)
```

```
0.8047244094488188
```

Next steps
This model was trained without any data preprocessing. Next step is to apply those data preprocessing to increase the accuracy. Also I want to try to train a model that is not pre-trained and see the difference in performance.

Colin Ching, (2019, March 27). *How to build an image classifier for waste sorting*. https://towardsdatascience.com/how-to-build-an-image-classifier-for-waste-sorting-6d11d3c9c478