DATA 2040 Final Project: Trash Classification

Harry Chalfin, Connor Uzzo, Alex Wang, Shuya Zhang
(Dated: April 2021)

Abstract

We seek to build a deep learning model to classify trash, separating out various types of recyclables from waste. In doing so, we apply a variety of convolutional neural networks (CNN's) to two independently collected photographic datasets of trash items. We find that the model can correctly classify trash items with an accuracy of above 90%, but only when the trash items fed to the model have a generic, blank background. This means that in building a successful trash classification model, it is important to first remove the background from the image so that the model can focus exclusively on each item in question.

INTRODUCTION

The problem of sorting trash efficiently and accurately is a serious environmental issue. Every year in the United States, some 254 million tons of trash are produced, or nearly a ton per citizen. This trash goes into landfills where it takes a very long time to decompose and disintegrate, so it is vital that as much of it as possible get recycled. If humans produce more trash each year than nature can reabsorb, our existence as a species will not be sustainable in the long run.

However, any non-recyclable trash which gets grouped in with the recyclable materials will make the overall recycling process less efficient and effective. Therefore both the sensitivity and the specificity of the selection of recyclable materials are important factors in any sorting method, whether performed by a human or by artificial intelligence. And of course, sorting trash is not a desirable job for workers, and so it would be nice if a machine could take over the job to free up the worker to perform some more critical task.

In this project, we build a deep CNN model to classify images of trash as either glass, metal, paper, plastic, cardboard, or trash. The classes in our dataset are relatively balanced so we will focus on simply trying to attain the highest test accuracy score possible.

One serious challenge we face is training an AI model to detect trash in a variety of conditions and environments. It should go without saying that neither the condition nor the environment in which an item of trash is found should have any bearing on its proper disposal method. However, because our goal is to train the model *based solely on image detection*, such a problem must be taken seriously. For this reason we ultimately apply our model to a separate set of images called the Trash Annotations in Context (TACO) dataset to judge its effectiveness on an independent test set. [1]

DATASETS

Thung-Yang Dataset

Our first dataset comes from Gary Thung and Mindy Yang [2], who collected a set of 2533 photographs of pieces of trash for their original trash sorting project. This dataset contains six categories: cardboard, glass, metal, paper, plastic, and trash. One distinctive trait about this dataset is that the image backgrounds contain shade but are rather clean and blank. All samples have the same image size and there is only one object in each image with no deformation. Each object takes most of the area of the image.

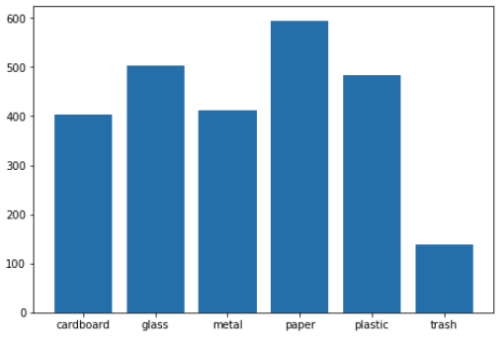

FIG. 1: Categories in Thung dataset

FIG. 2: Sample image from Thung dataset

TACO Dataset

One problem is the sample size of the Thung dataset is too small, and so we wanted to expand it. We found the TACO (Trash Annotations in Context) dataset, which is based on the COCO (Common Object in Context) project. TACO is an open image dataset of waste in the wild which contains photos of litter taken primarily by mobile phones under diverse environments. COCO [3] is a large-scale object detection, segmentation, and captioning dataset. The dataset has 60 categories (see Figure 3) instead of 6 as shown in the bar plot, so we mapped them into the original 6 labels. (See Figure 4.) Different from the clean background in the Thung dataset, images in the TACO dataset have a more complicated and realistic background (see Figure 5) and the distribution is shown in the pic chart. Another difference is that in the TACO dataset, one image can have multiple objects within. There are 1500 images in the TACO dataset with 4782 segmentations. Once we combine two datasets, we will have 7315 samples in total.

Background Cleaning

To keep two datasets similar, we decide to try removing the backgrounds for the TACO dataset. (See Figure 6.) The TACO dataset comes with a bounding box for each segmentation. We extract the segmented object from the original image, transfer it into RGBA arrays in order to make the background transparent. Then we paste it to a new image of the same size but with white background. The background color is subject to change. Finally we convert it to RGB to save it as a jpg file with optimization to reduce the image size. The entire dataset is 403.9MB while the original TACO dataset is 2.63GB. The advantage in using such a motley collection of photographic data is that it forces the model to generalize well and learn about the items of trash themselves, not about their surroundings.

As we've mentioned earlier, one image can have multiple segmentations. The following sample (see Figure 7) perfectly illustrates this point. There are 10 segmentations in this single image which fall into 4 different categories.

RESULTS AND ANALYSIS

Model 1: Inception on Thung

The first trash classification model we built used the Inception ResNet V2 model as a base, and it was trained on the Thung dataset. This dataset contains images of individual pieces of garbage on white backgrounds, classified as either cardboard, glass, paper, metal, plastic, or trash. This was our most successful model, with an accuracy of between $89-92\%$ on most runs. We believe this was most likely due to the blank backgrounds, which made it easier for the model to recognize the object to be classified in each image.

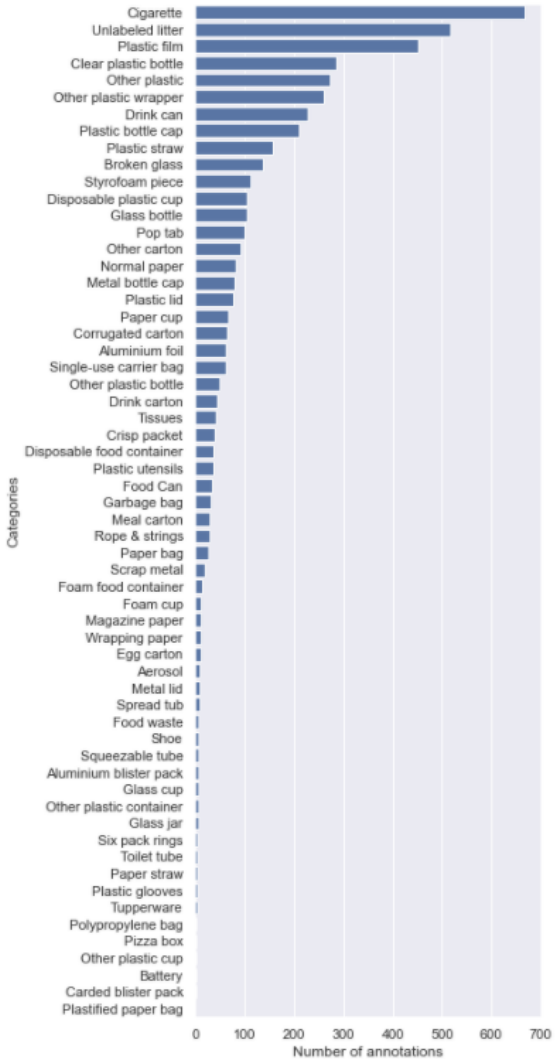

FIG. 3: Categories in TACO dataset

The choice to use Inception ResNet V2 was made after it out performed VGG16 and standard ResNet.

Inception networks are a class of CNN's that run convolutions with multiple differently sized kernels on each convolutional layer, concatenating the results. This allows them to recognize images of different sizes very well, which likely led to the modelâs strong performance on this dataset. In versions of Inception ResNet, this inception quality is applied to the standard residual

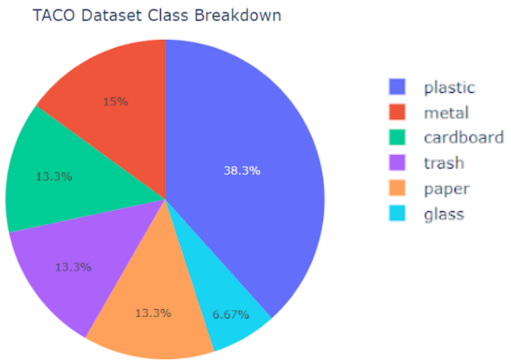

FIG. 4: New categories in TACO dataset after mapping

FIG. 5: Sample image for TACO dataset

blocks used by ResNet, which try to calculate the model's approximate error in output rather than the output itself. This is done using skip connections, which help the model ignore non-useful information. The combination of the inception quality and the residual blocks allow the model to be both highly effective and relatively efficient, making it a very powerful neural network. After initializing Inception ResNet V2 as a base, we simply adding a single global average pooling layer followed by a softmax-activated dense layer of 6 nodes to make our head model. Including extra layers in this head creates the risk of forming an information bottleneck, so we decided to keep it as simple as possible.

We trained our model in two stages, leaving the base model parameters unfrozen for the full duration. Both stages used exponential decay learning rate schedulers and early stopping callbacks, which were found to be more than sufficient for mitigating overfitting. We originally started training with an initial learning rate of $1 \cdot 10^{-4}$, however, it was ultimately found that bringing this down slightly to $7 \cdot 10^{-5}$ led to an increase of $2-3\%$ in accuracy. The learning rate scheduler in this stage was

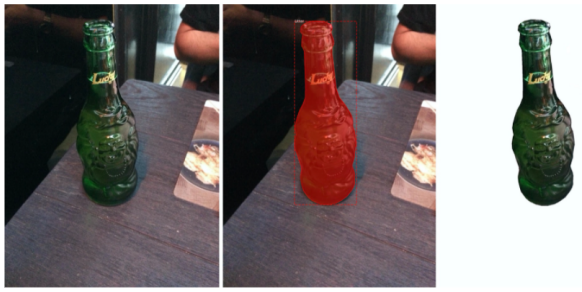

FIG. 6: Sample for background removing with single segmentation

FIG. 7: Sample for background removing with multiple segmentation

set to decay by 94% in 5000 steps. This first stage was allowed 20 epochs to train, though early stopping would usually kick in before all 20 were run. The early stopping callback returns the model to the weights that resulted in the lowest validation loss after the first stage of training is complete, and we set it to stop the model training if the validation loss does not decrease for 5 epochs. After this first stage of training ends, the model is fine tuned by training for another 15 epochs with the second exponential decay learning rate scheduler, this time starting at $7 \cdot 10^{-7}$. The decay steps are also increased to 10,000, making the decay in learning rate a slower process, and the decay rate is increased from 94% to 96%.

In the loss history, the point in which the second stage of training (fine tuning stage) starts can be clearly seen, as it is the point where the loss becomes much less noisy and levels off (around epoch 13).

Model 2: Inception on TACO with background

Following the success of our first model, we wanted to see if we could train a model to classify images of trash that were taken with random backgrounds. We felt this would be much more useful in the real world, since trash is rarely found against a blank backdrop. The thought was that we could use transfer learning

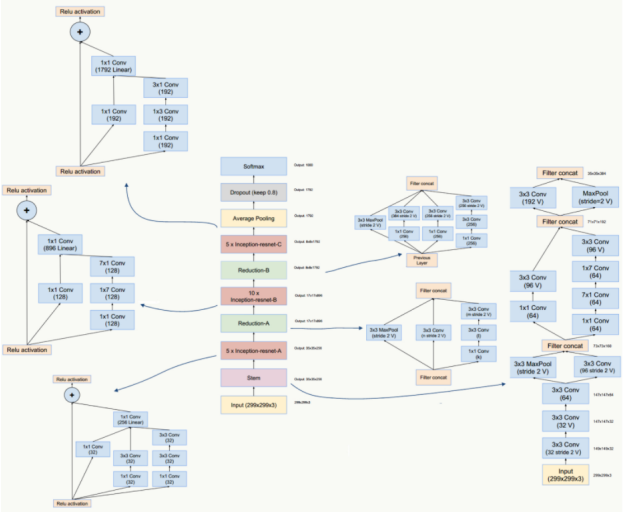

FIG. 8: Inception ResNet V2

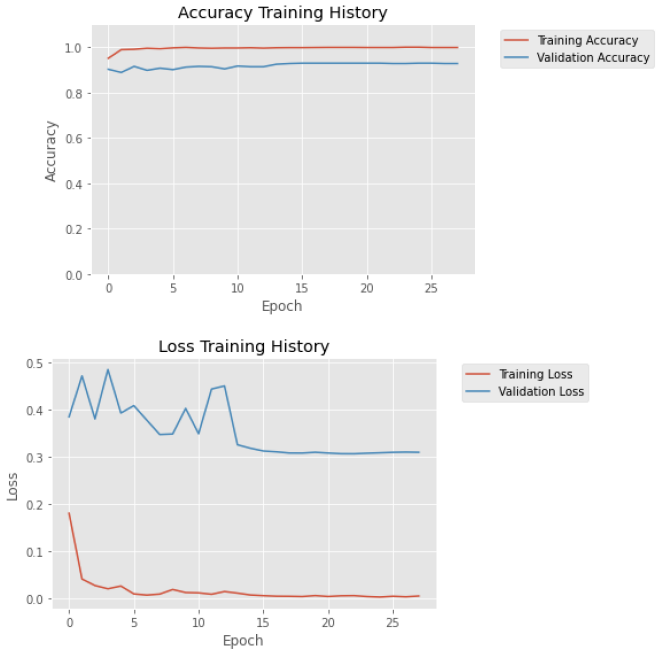

FIG. 9: Accuracy and loss training history for Model 1

on the first model we built to adapt it to classify trash in varying settings. We trained this new model on the modified TACO dataset, which we altered to have the same six classes as our first model. This dataset had around 4,800 images, so we had to build a data generator object to train the model without depleting too much RAM. The breakdown of classes was also much different than Thung's dataset, with plastic being by far the largest class and glass being the smallest.

We were surprised to find that this model was far less successful than the first, only achieving validation accuracy scores in the $55-65\%$ range.

FIG. 10: Accuracy and loss training history for Model 2

To see how effective our transfer-learned model is compared to one that started on the TACO dataset, we built another model which we will call our "fresh" model, again using Inception ResNet V2 but this time training on the TACO dataset from the start. This model ended with slightly better results than the transfer-learning model, with accuracy between $65-70\%$. Still, this is about 20% lower than the model trained on Thung's dataset. Based on these graphs, we were able to draw the conclusion that the randomized backgrounds of the TACO dataset made the images much harder for the model to classify.

Model 3: Inception on TACO without background

Before merging the TACO and Thung data, we first tried to build an Inception model on the TACO images with their backgrounds removed. We built the Inception model in the same manner we did to the Thung dataset, with exactly the same settings including the learning rates, early stopping, and others.

After 6 epochs, we got a validation accuracy of 57%, which is not as good as the model trained and tested on the Thung dataset but similar to our result on the TACO data with the backgrounds. By looking at the training history, we found that the training loss is bouncing around and the validation accuracy is significantly below that of the training, which is 71%. Therefore, we believe there exists the problem of overfitting here. The Inception model performs drastically different lyon

two datasets and this divergence may suggest that there could be some problem regarding the TACO images with their background artificially removed. We will keep this concern when working on the merged dataset.

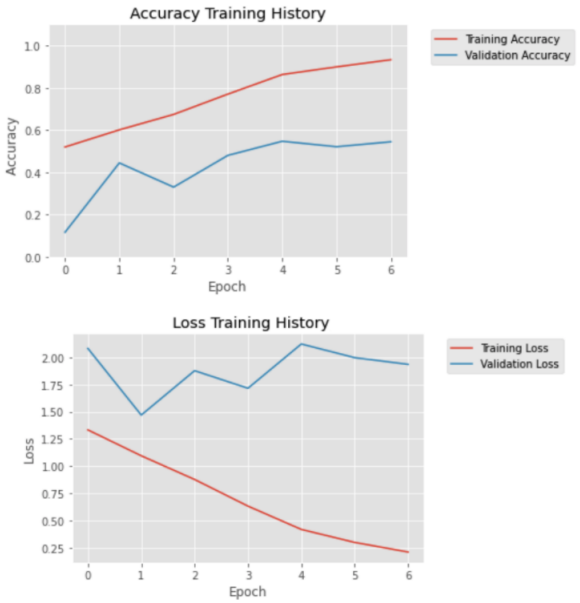

FIG. 11: Accuracy and loss training history for Model 3

Model 4: Inception on Thung and TACO without background

Next, we began to work on the merged dataset of Thung and TACO, which contains 7315 images in total. Because the TACO dataset is very imbalanced with a third of it being plastic items, the combined dataset is also slightly imbalanced. We made sure that the data are well-mixed regarding both the two datasets and six categories, and used 75% of the data as training data.

After merging the dataset, we decide to apply the inception model to the merged dataset. The parameters and settings are the same with our Model 1 and Model 3, which gives us a validation accuracy of 65% and the training accuracy of 97%. According to the accuracy and loss for the training history, the training loss decreases at the beginning then it begins to stay the same and bounce around. The gap between training and validation accuracy and the performance of the validation loss suggest the potential problem of overfitting again.

To solve the problem, we decreased the learning rate to $7 \cdot 10^{-7}$, which gives us a better validation accuracy of 70% but the overfitting problem is still present and we can see that the model is not improving.

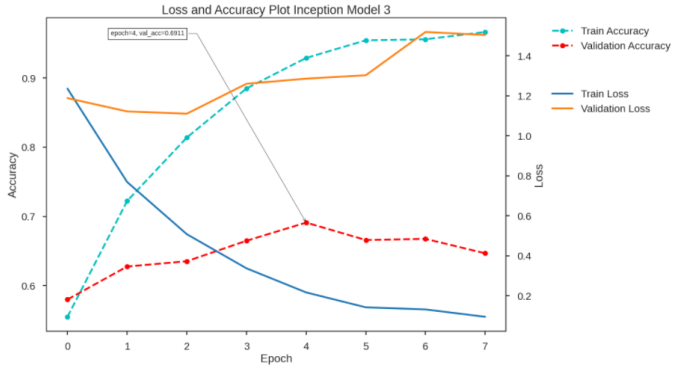

FIG. 12: Accuracy and loss training history for Model 4 (Inception Model 3)

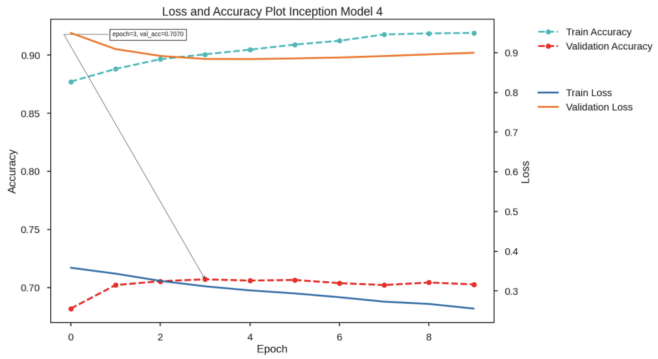

FIG. 13: Accuracy and loss training history for Model 4 (learning rate tuned)

Solutions for overfitting and next steps for Model 4

Since the parameter tuning was not helpful, we came up with three hypotheses of potential reasons why this may have occurred. First, the data needs to be mixed well. To make sure the merged dataset is well-mixed, we checked the positional distributions of each category. However, this was not helpful.

Second, the TACO data is imbalanced, with about a third of the objects classified as plastic. We tried applying SMOTE (Synthetic Minority Oversampling Technique), an oversampling method. We also changed the class weights during our training process, hoping to give more weight to the categories with less presence. However, this was not helpful.

Third, we also tried to adjust the model structure in three aspects. First we tried adding dense layers, second we tried adding dropout, and lastly we tried freezing the base then unfreezing it after training once. Unfortunately, none of these worked.

Despite the efforts, we think there are some next steps that can be taken to tackle this problem. For example, changing the background color from pure white to a color that is more similar to real-life white walls.

FUTURE DIRECTIONS

Much of our model's misclassifications involved classifying plastic as trash, or vice-versa. (See Figure 14.) Plastic and glass were also frequently confused, particularly in the TACO dataset. Both of these errors are understandable, as plastic can be opaque and dirty looking like trash, or translucent and shiny like glass. However, we would ideally like to find a model that can differentiate these classes better, especially because most plastic is recyclable and trash is not. Such a model would probably have to be more focused on fine details in each picture than our original model was, which could be accomplished by increasing the resolution of each input image. This would allow for convolutional layer kernels that have smaller receptive fields relative to the complete image, which could possibly recognize fine details. These fine details, like the roughness of scratched-up plastic or the stronger shine of glass versus plastic, would most likely be the best way to try and improve our accuracy. Inception ResNet V2 would still be a very effective model in this case, though it would take significantly longer to run due to the larger size of the higher resolution images.

Also, our best performance by far was conducted on the Thung dataset, which as we mentioned earlier contained only images with blank backgrounds. All other models, however, performed significantly worse. We could possibly have created a stronger model if we built an even larger dataset of images with more diverse backgrounds, as this would force the model to focus less on the surroundings and more on the piece of trash in the image. We would probably want to find multiple other datasets to do this, so that we have heavily generalized images of trash in all different places. A more diversified model would theoretically be able to classify trash effectively regardless of its background, which would be much more useful in practice than a model that only classifies trash well on a single background.

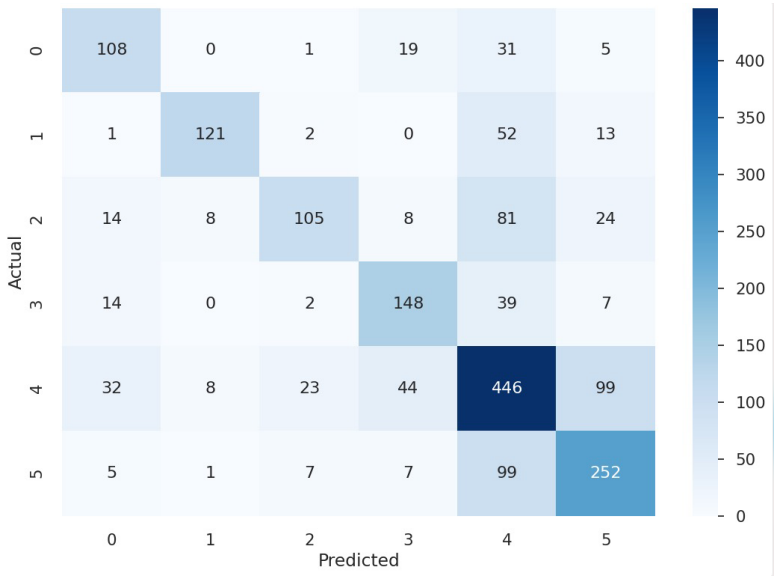

FIG. 14: Confusion matrix for Model 4. Note that plastic easily gets confused with trash.

CONCLUSION

There are several key takeaways from this project. The first and perhaps most important is that it is in fact possible to build a deep learning model to sort trash with a high level of accuracy. This kind of work then paves the way for robots to do trash classification. Second, we find that classifying images of trash on blank backgrounds is easier than on random backgrounds. This means that a good trash classification algorithm should work by first isolating the trash items from their surroundings and only then identifying them in the proper disposal category. Lastly, we found that the transfer learning model performed about as well as the model trained only on new data. This gives us confidence that this model can be easily generalized to a wide variety of samples, enabling us to base AI technology directly off this work.

[1] P. F. Proenca and P. Simoes, "Taco: Trash annotations in context for litter detection," (2020), arXiv:2003.06975 [cs.CV].
[2] Q. G. Yuiji He and M. Shi, "Trash classification using convolutional neural networks project category: Computer vision," (2020).
[3] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. DollÃ¡r, "Microsoft coco: Common objects in context," (2015), arXiv:1405.0312 [cs.CV].
[4] D. Quinn, "How do i read image data from a url in python?".
[5] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of resid-

ual connections on learning," (2016), arXiv:1602.07261 [cs.CV].
[6] A. Hobbs, "Why you could soon have a neural network on your smartphone,".
[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, J. Artif. Int. Res. **16**, 321â357 (2002).