Using a Convolutional Neural Network to Differentiate Recyclables From Ordinary Trash

Matthew Vanlandingham Stephen Smith Cory Nelson Ryan Armstrong Richa Phulwani

***Abstract*—This project had the goal of developing a neural network that could identify different types of recyclable materials as well as trash because of the growing need to have automated sorting mechanisms for the massive amounts of recyclable waste produced by humans. The project approached this by using a branching convolutional network and resizing the original images from 512 by 384 pixels to just 100 by 100, theoretically keeping training times low while still providing strong classification results. The data used was found from a previous study of the pursuing similar classification goals, consisting of images of six different categories: glass, metal, plastic, cardboard, paper, and trash [1]. The network took the randomized input data and ran it through a series of branches designed to split it into grayscale and color images, before recombining them and attempting to identify patterns amongst each of the produced variants of the input images. The results indicated a 61 percent accuracy on the testing data using images with a resolution of 100 by 100 pixels. This did not improve on the accuracy of the network that the dataset was retrieved from, which was 75 percent, but it did well considering the low resolution image inputs compared to the large image sizes the original network used [1]. Benefits seem to be best realized by putting further development toward maximizing low resolution detection for faster training and processing.**

I. Introduction and Background

The identification and then categorization of recyclable waste is a pressing concern for recycling centers. Among the most severe strains on the efficiency of recycling programs is the contamination of materials passed on to recycling plants. The contamination rate of recycled materials in the United States is between 15-25

In producing a solution to the recycling sorting problem, our team decided to build a convolutional neural network, because this problem required a network able to process many images, and then categorize those images based on a distinct set of features for each category, and convolutional neural networks are optimal for detecting features for each category of images. In order to develop the neural net, a significant number of images were gathered and then used to train the network to identify which category a new recyclable material would belong to. Specifically, our neural network is intended to be able to identify items belonging to the following categories: Glass, Paper, Cardboard, Plastic, Metal, and Trash.

A similar project was found with the same goal of identifying recyclable materials using a convolutional neural network. The dataset was used from this project to train this network, as it contained a substantial amount of material with over 2000 images of recyclable materials and garbage [1]. The project was also used as a benchmark to measure the effectiveness of this project, with a goal to reach a higher testing accuracy rate.

II. Methods

Dataset: The dataset was taken from an open-source repository of a project similar to ours. The people who had made the dataset had taken pictures of plastic, glass, metal, paper, cardboard, and trash materials on their own. Each category of material had different objects in it, such as plastic bottles, plastic jugs, plastic containers, etc for the plastic category. The categories had 482, 501, 410, 594, 403, and 137 number of pictures, respectively.

The various networks tried were visualized with the SVG.

The process began with a similar convolutional network setup as used for classifying the MNIST data set. Due to the greater complexity of the images in the dataset, additional changes were made to the basic network in order to improve performance. One of the first components of building the net was simply preprocessing the data correctly by loading them into numpy arrays after resizing them down to 100 by 100 pixels to speed up progress instead of processing the images on each attempt. The images then had mean subtraction performed so that the intensity values of the channels would perform better with activation functions such as ReLU [4].

Each type of material had class labels associated with it that were converted into one-hot encodings. This allows it to function with the previously built networks, and give visualization of what the network thinks of an image by showing the percentage likelihood of each category to which it believes a certain image belongs. The images were finally taken shuffled among each category and the array contents split in half, such that half of the images would be used for training and validation while half would be used for testing purposes. This ensured that images near each other in the original images were dispersed, as some were the same object at different angles or rotations. The entire training set itself was then shuffled in order to prevent any sort of catastrophic forgetting from occurring by only training on one image type at a time.

Changes to the network began with the idea of splitting it into separate trees with the hope of reducing time spent processing through parallelization [5]. Some of the first few models took the approach of differently sized kernels to identify different types of patterns. The first model had a pair of branches in which the first brach identified patterns in horizontal rectangular sections of pixels and the next branch

did the same for vertical sections. A modification to that was a model with two branches: one that identified smaller details by using a small kernel size and one that identified larger details with a larger kernel size. Ultimately the current network was settled upon after testing out additional ideas. The network is best explained by detailing four different parts.

The first part is an interesting component that adds gaussian noise to the input in order to tilt the network towards trying to identify more general features of the images instead of very specific components that would tend toward overfitting [6].

The second part takes that adjusted input and pools it to reduce the number of parameters down to 50 by 50, and proceeds to split it into the first pair of convolutional layers, the first of which converts the images to grayscale to identify color independant features, while the other retains the color of the images [7]. Each of these are passed through two layers of convolution and proceed to the next section of the network.

The third section of the network pools the previous two branches and splits the results off into three additional branches [8]. Two of the branches are continuations of the previous grayscale and color convolutional layers. The third branch in the middle is new as it concatenates the two poolings from the grayscale and color branches and combines the information to find additional patterns. All of this is then used as the network passes into the fourth section.

The final part of the network is fairly simple and similar to any other convolutional network. The final components of the previous three branches are pooled and concatenated once again before going through an additional convolutional layer followed by a two-dimensional spatial dropout. A regular dropout randomly sets a fraction of input units to zero at each update during the training time, which helps prevent overfitting. A spatial dropout works similar to a regular dropout but differs by dropping entire feature maps instead of singular units [9]. This is then flattened and passed through several dense layers with dropout to help reduce overfitting before producing the final output.

These outputs can then be visualized with a percentage reading on how likely the network thinks the image belongs to that category. When compiling the model the Adam optimizer was used with a learning rate of 0.0001 and the amsgrad value set to true so that the network better remembers older gradients and continues to keep a working memory of each type of material [10].

III. Results

After running the network through 200 epochs with a batch size of 200, there was noticeable gain in the overall accuracy of the model reaching a peak testing accuracy of 61

IV. Discussion

So far there is no capability to improve the actual accuracy beyond that described in the original Trashnet.

Since the dataset contained different types of objects for each category, in the future, the neural network could be trained on a dataset that is more specific for each category.

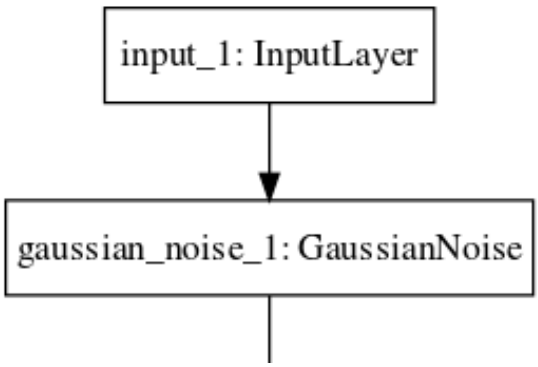

Fig. 1. Section one

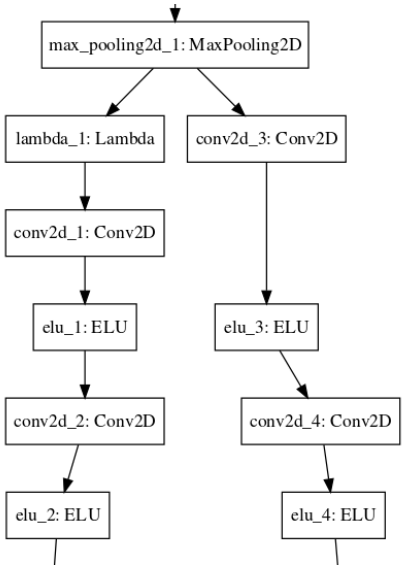

Fig. 2. Section two

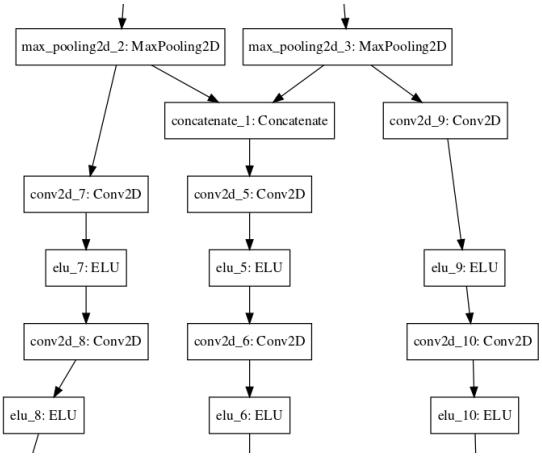

Fig. 3. Section three

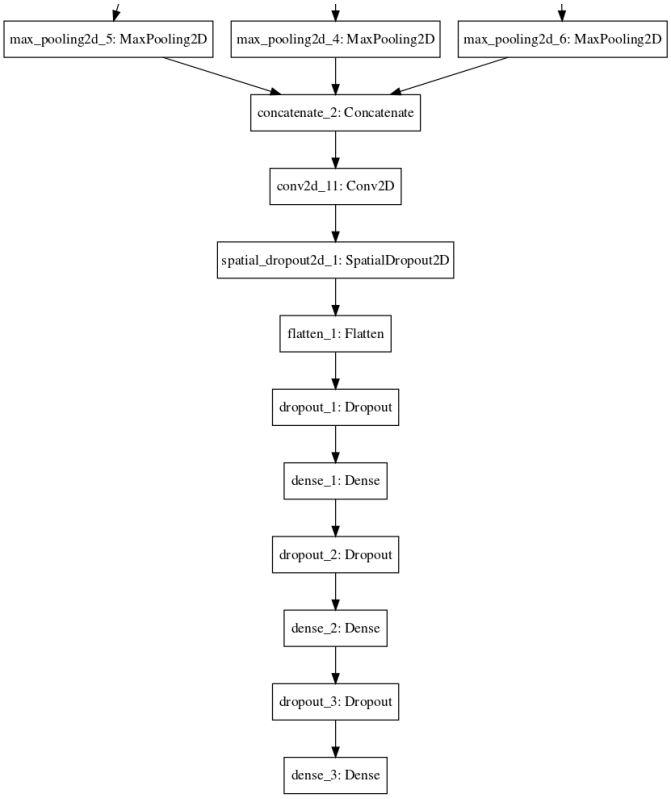

Fig. 4. Section four

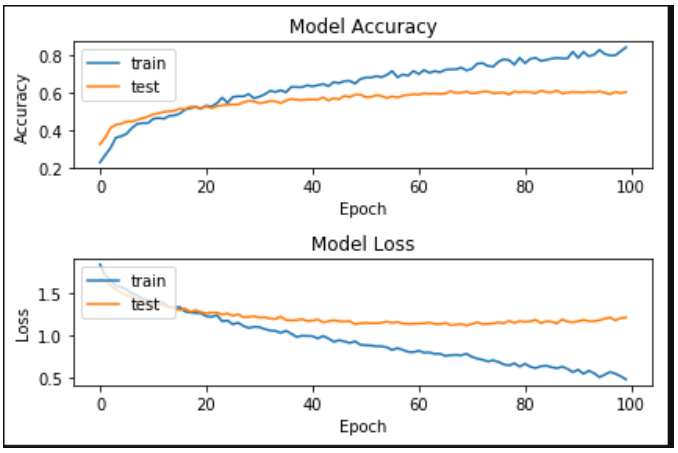

Fig. 5. Model Loss and Accuracy over 100 epochs

It would be easier to train the network with specific types of objects since they would all look similar to each other. For example, some of the more specific recyclable categories could be: plastic bottles, glass bottles, and aluminum cans. There are many other recyclable materials that the project could expand on.

Secondly, due to an oversight for approximately the first third of network attempts created, the input data was being shuffled incorrectly, possibly resulting in catastrophic forgetting. As a result, these network models should be revisited in the future with adjusted data shuffling methods in order to see their viability. Attempts were made to produce a network which could potentially run faster, have shorter training times, and produce reasonable results on the level of or better than a single path network.

There was no comparison to the original trashnet training speed due to it running on a different backend and being written in Lua; however, one major comparison point between the networks is that the input resolution of the trashnet images was 512 by 384 as compared to the resized 100 by 100 used as the input to this project [1]. In regards to this, a very satisfactory outcome was gained being able to identify recyclables with at least 60

Additional investigations into this network are valuable, especially because being able to recognize lower resolution images could potentially allow filtering of larger quantities of recyclables simultaneously. Overall this would improve efficiency due to the simple quantity increase that would be able to be assessed at a time.

V. Methods

Preprocessing (resizing, shuffling)
Bottleneck
Batch Normalization
Squeeze
Spatial Dropout
Activations, Learning Rates, Kernel Sizes

[1].

References

[1] C. Miranda and F. Von Zuben.

References

1. https://github.com/garythung/trashnet Thung, Gary, and Yang, Mindy. Dataset of images of trash; Torch-based CNN for garbage image classification. Github. April 9th, 2017.

2. https://www.kansas.com/news/special-reports/article1078183.html Sylvester, Ron. What Waste Connections does with your unsorted recycling The Wichita Eagle. November 18th, 2011.

3. https://www.satyenkale.com/papers/amsgrad.pdf Reddi, Sashank, et al. On the Convergence of Adam and Beyond. International Conference on Learning Representations. New York. 2018.

4. https://becominghuman.ai/image-data-pre-processing-for-neural-networks-498289068258 B., Nikhil. Image Data Pre-Processing for Neural Networks. Becoming Human: Artificial Intelligence Magazine. September 10th, 2017.

5. https://arxiv.org/abs/1511.02954 Miranda, Conrado, and Von Zuben, Fernando. Reducing the Training Time of Neural Networks by Partitioning. School of Electrical and Computer Engineering. University of Campinas, Brazil. January 3rd, 2016.

6. https://stackoverflow.com/questions/37020754/how-to-increase-validation-accuracy-with-deep-neural-net How to increase validation accuracy with deep neural net? Stack Overflow. May 4th, 2016.

7. https://www.johndcook.com/blog/2009/08/24/algorithms-convert-color-grayscale/ Cook, John. Three algorithms for converting color to grayscale. John D. Cook Consulting. August 24th, 2009.

8. https://arxiv.org/abs/1411.4280 Tompson, Johnathan, et al. Efficient Object Localization Using Convolutional Networks. New York University. June 9th, 2015.

9. https://keras.io/layers/core/ Core Layers. Keras Documentation.

10. https://www.satyenkale.com/papers/amsgrad.pdf Reddi, Sashank, et al. On the Convergence of Adam and Beyond. International Conference on Learning Representations. New York. 2018.