Springboard–DSC
Capstone 3
Computer Vision in Convenience Retail
By Dennis Francis
December 18, 2023

(1) Introduction

An image with annotation of a 3-foot confectionary section

The inception of this project came from a merchandising problem experienced in the convenience retail store business. The footprint of a convenience store is methodically laid out and planned. Each three-foot section of shelf space is given a label and merchandised according to a schematic. The goal of this project was to build a computer vision model that can classify the merchandising category of an image, as in Figure 1. This is an incredibly difficult challenge to solve when one considers the vast array of features that can set a section of merchandise appart. A typical convenience store may contain over 2,000 products. Nevertheless, we hope that utlizing a pretrained model trained on millions of images shall be capable of generalizing well for our purposes. Out of the three models tested on two separate datasets, the model that was selected in this study will have applications in retail and merchandising. We believe it would be of interest to companies in the retail automation space, such as Amazon Go and grabango, and ShelfEngine.

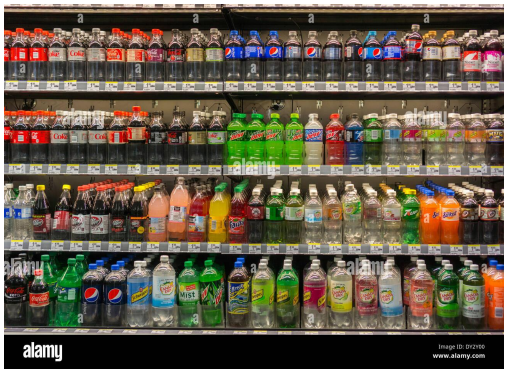


Fig. 1 - We seek to identify a model that will classify this target image as "drinks".

There is a real need for being able to identify merchandising categories. With over 1 million convenience stores worldwide, our ultimate goal is to pave the way for making this technology open source and available to independent operators who can expand the project and implement their own systems. Currently, searching "computer vision retail" on GitHub results in only 47 repositories, with most having little engagement. We believe that given enough time, data, resources and dedication, an independent convenience store operator can develop their own retail system that automates retail inventory management, thus improving lives and generating more profit.

(2) Approach

(2.1) - Data Acquisition and Wrangling

Transfer learning is a relatively recent phenomenon in deep learning that has made the field more accessible to entry-level practitioners. Instead of needing millions of images to train a model from scratch, one can download a csv file containing the weights of a model that was already trained. One can take the model and retain it, only removing the last layer (called the "head") and repurposing the model for a new problem, sometimes entirely different from the original task the model was trained for. Using a pretrained model means that one can achieve incredibly good results with only a few images–say, a few hundred up to a few thousand or so (compared to millions) .

In this project, we scraped images of convenience store images from the web. We used duckduckgo-search[5], which is a function that takes two main arguments: 1) a google-like search term such as "convenience store" and 2) the number of results, say 10. So, given the arguments "convenience store" and "10", the application will return 10 urls of images of convenience stores. Images were searched, labeled, and stored in a dataset using Python code. The image categories searched for were: 1) cigarettes; 2) candy; 3) drinks; 4) ice cream; 5) chips; 6) other. After downloading and organizing the files, the original dataset contained 759 images organized in the following categories:

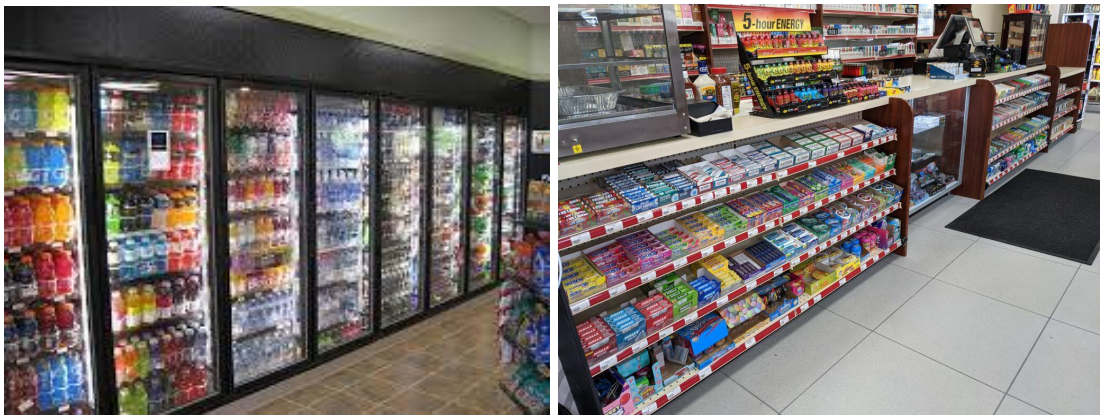
cigarettes: 133 images
candy: 120 images
drinks: 125 images
ice cream: 113 images
other: 152 images
chips: 116 images

Fig. 2 - Labeled images of ice cream, cigarettes, and drinks from dataset1_train

Because the images were downloaded directly from the web, naturally there were some blatantly incorrect images that slipped into the dataset. See Figure 3, containing images that are irrelevant. Although these photos are technically related to chips, cigarettes, and drinks, they are not relevant to the problem we are tring to solve. For this reason, a second dataset was created by removing these images.

Fig. 3 - Irrelevant images. On the left is a t-shirt alluding to the "chips" category. In the center is a t-shirt with a logo of a popular American brand of cigarettes. On the right is an image of coolers for holding cold drinks.

In addition to irrelevant images, there were some images that contained too much information. Observe the following images of drinks and candy:



.

These images contain too much information about features we don't want the model to learn. For instance, in the candy image on the right, observe how there are 5-hour Energy drinks on the sales counter and even cigarettes far off in the background. These features could confuse the model and lead to poorer results.

Although there has been much research into understanding fully the inner workings of neural networks, it is still appropriate in practice to treat them as a "black box." In this regard, training a neural network is more of an art than a science, and with this line of reasoning, we believe it would be best to eliminate extraneous features by simply cropping them out as shown in Fig. 4.

Fig. 4 - Manually cropping an image so that the neural network will be less confused as to what constitutes "drinks"

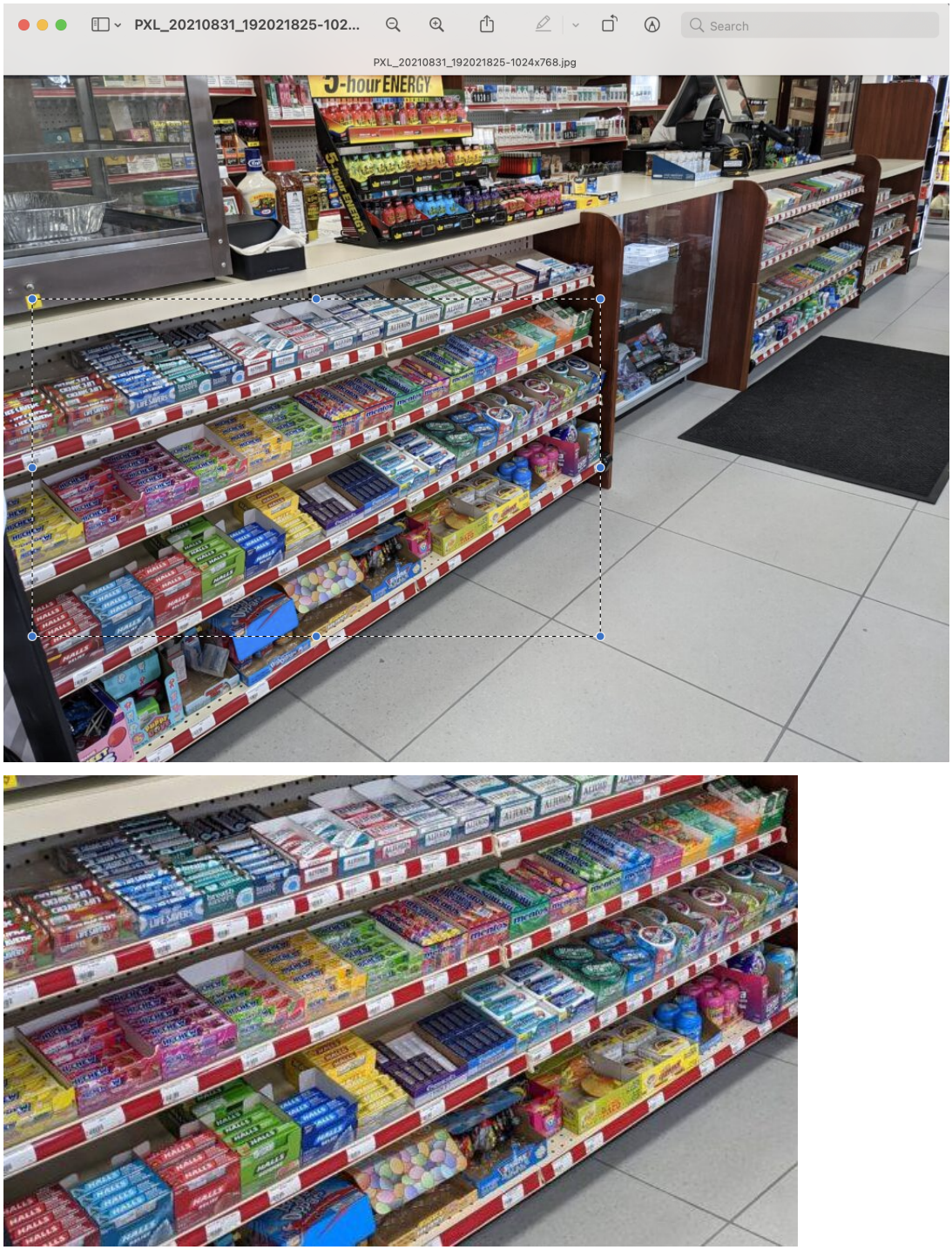


Fig. 4.1 - Manually cropping an image so that the neural network will be less confused as to what constitutes "candy"

Cropping images using the default MacOS image editing software (Preview) took roughly 2-3 hours on the original 759 images.

After manually cropping and removing irrelevant images, the dataset had only 668 images remaining. This naturally raised some concerns of not having enough data to train a model. Furthermore, the classes were imbalanced which is also not good. See Figure 5, where 'candy' contains over 250 images and 29.3% of the data, while 'ice cream' makes up only 10.6% with barely one hundred images.

This problem was addressed by flipping images 180 degrees and randomly cropping them. See Figure 6, where an image of cigarettes is randomly cropped, resulting in 4 additional images. A custom function was written in python and can be seen in full detail in the [EDA and Wrangling notebook].

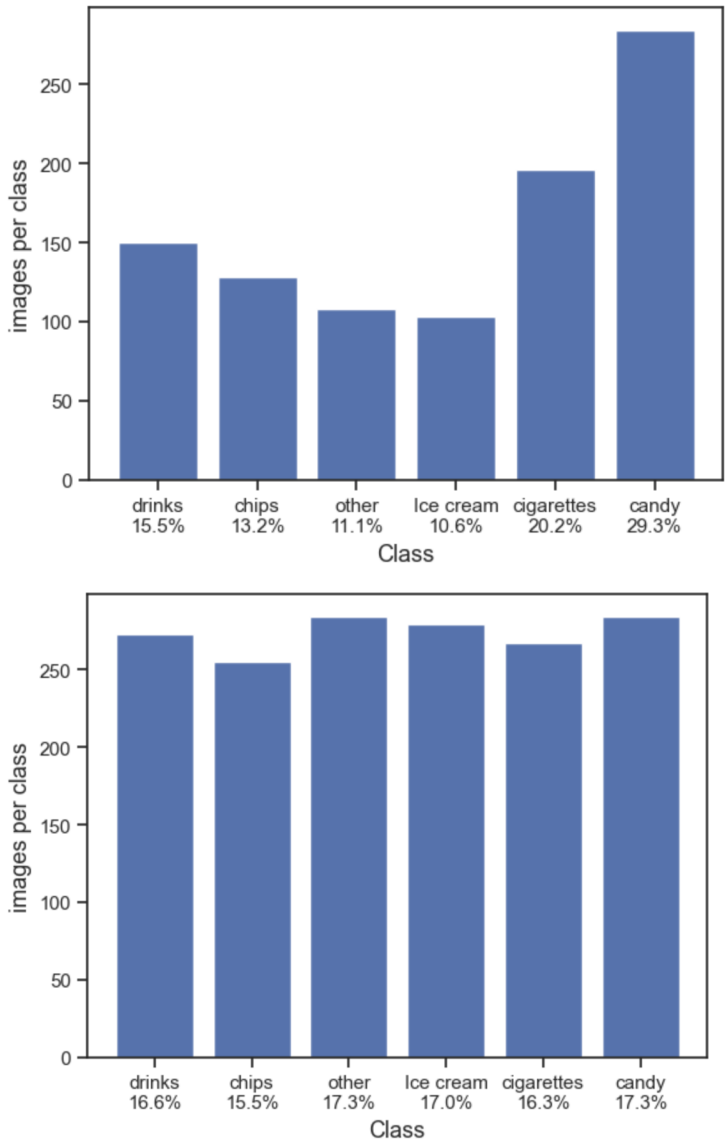


Fig. 5 - Addressing class imbalance after adding flipped random cropped images

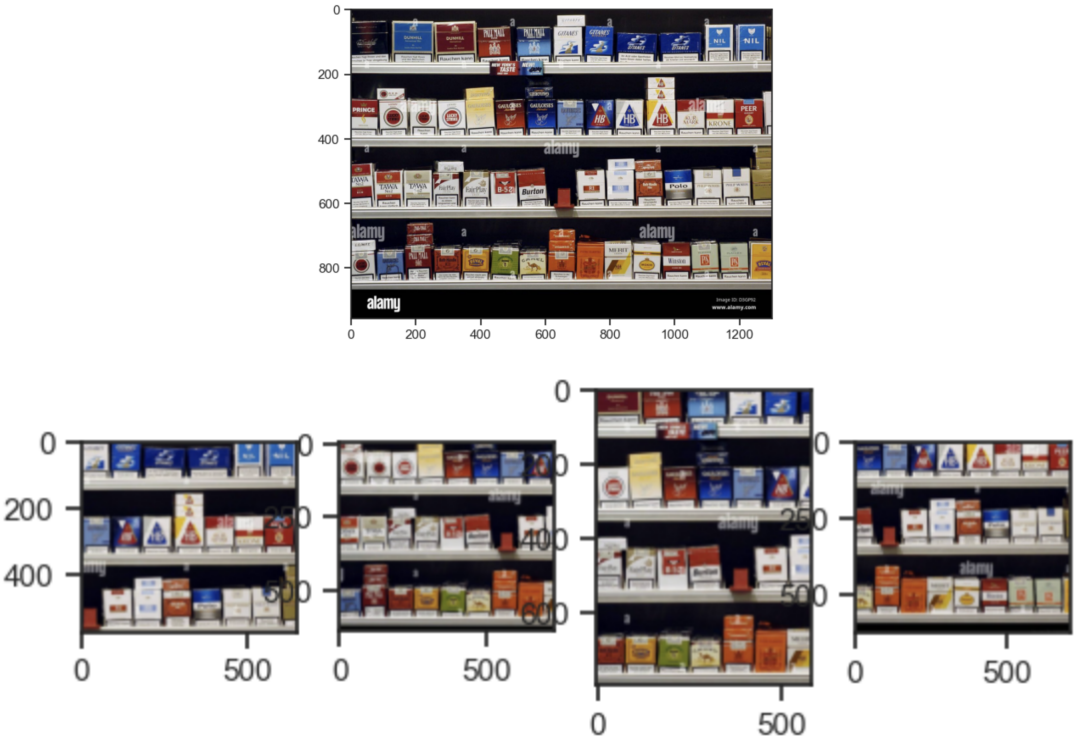


Fig. 6 - Random cropping of an image of cigarettes

(2.2) - Storytelling and Inferential Statistics

In Data Science and Business Analytics, we are normally given a tabular dataset with features and a target. Obtaining a predictive model is usually a straightforward task: using pandas and sci-kitlearn, we clean the data and apply a model such as logistic regression or a random forest. However, we cannot proceed with this framework for the problem at hand. How can we obtain a predictive model with data that seemingly has no features?

The answer is that we employ a deep learning model that will learn the features of the images. In general, a deep learning model such as a convolutional neural network will learn basic features like edges and gradients, and then, with subsequent layers, identify more features such as shapes, sizes, and in the final layer, classify using softmax. The combination of multiple layers and thousands of weights is what enables the network to classify.

Before diving into the training a model, it is important to explore the data. Because the data are images, we can conduct a brief analysis of the the size and dimensions of the images to gain insight into how the model will perform. Take for instance, an image of chips, seen in Figure 7 which contains 2,560,000 pixels. In order to avoid massive amounts of computation, we scale the image down significantly. In our model, images are scaled down to a square containing 50,176 pixels (224 x 224). This is more than enough for the model to learn the features required

for prediction. Observe in Figure 8 how us *humans* can still easily identify images of chips that have been scaled down to only 100 x 100 and 50 x 50 pixels.

Fig. 7 - A 1600 x 1600 image of chips

Fig. 8 - Two images of the same "chips" image, scaled down to 100 x 100 on the left and 50 x 50 on the right. Note how although the image on the right only contains 2,500 RGB pixels, we are still able to glean the features to identity it as an image of chips; we can see the shelf borders and outlines of what could be small, single-serve bags of chips.

The training dataset contains 1,321 images. As stated above, images can come in a variety of sizes, the most common being 224 x 224. The average image size is 518 x 465. The smallest image is 71 x 84, and the largest image is 5312 x 2988. The IQR of images lie somewhere between 224 x 224 and 650 x 536. Observe in Figure 9 that most images have a width between 200-pixels.

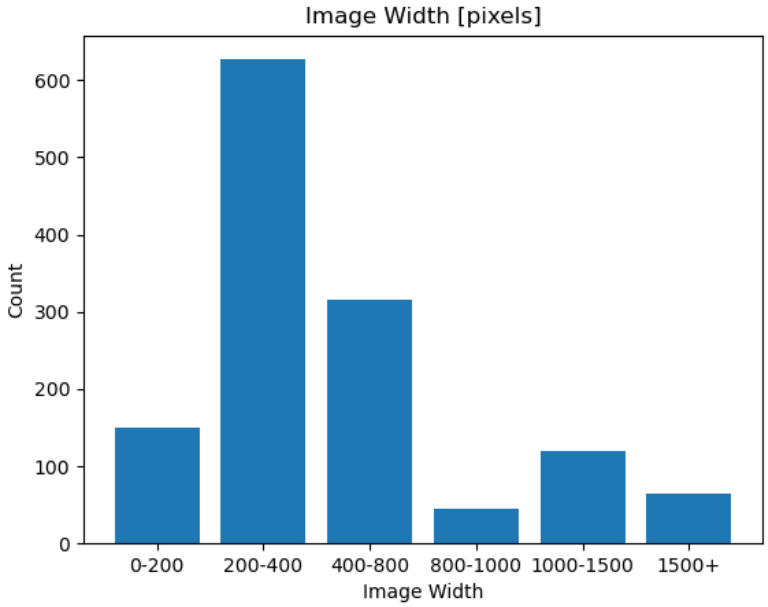


Fig. 9 - Images between 200 and 400 in length occur most frequently

We can also analyze photos based on their pixel values. Images have three color channels ranging in pixel values of 0 and 255, with 0 being black and 255 being bright. We can take an image and plot the distributions and, compare the distribution with that of another image (see Figure 10).

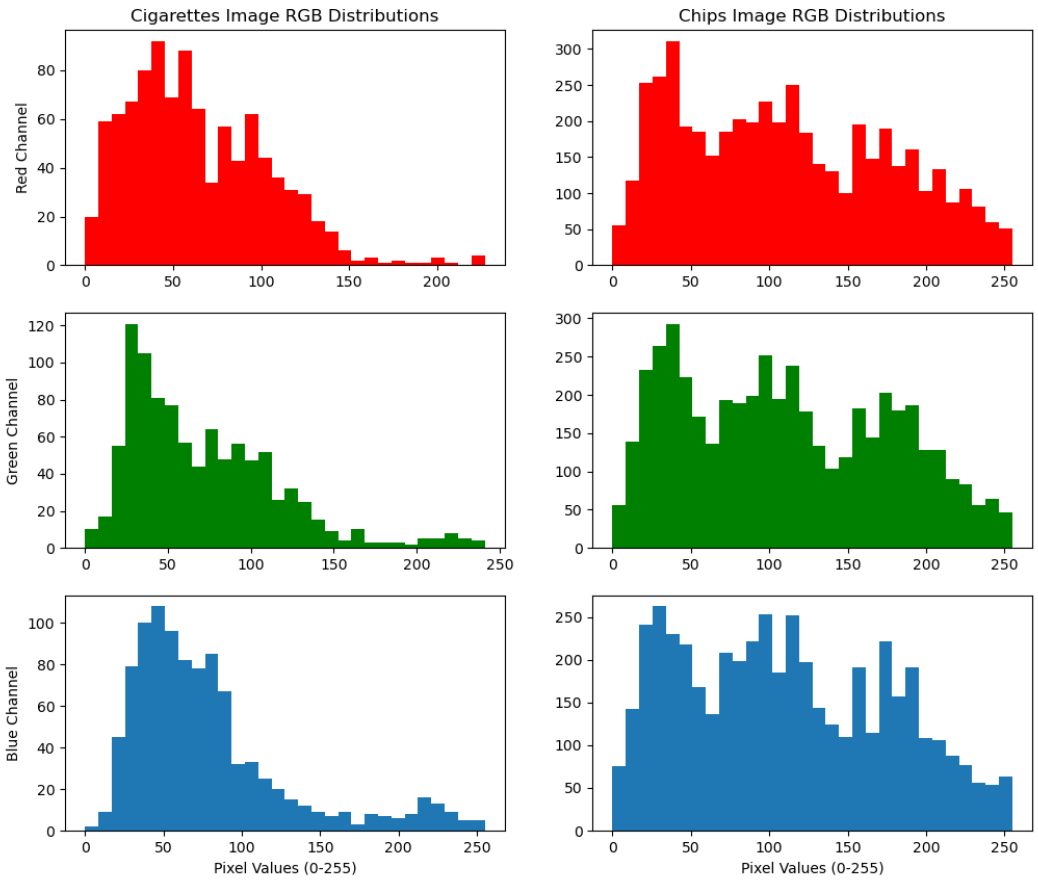


Fig. 10 - A side-by-side comparison of the RGB pixel distributions of chips and drinks photos

(2.3) - Baseline Modeling

ResNet50 was applied as our baseline. The neural network was trained in 3 epochs with a learning rate of 0.1 and cross-entropy loss. The accuracy achieved was 48%, which was rather alarming given that this model architecture performs well in many other applications.

Baseline: ResNet50 on dataset 1:

```
Accuracy: 0.4822
              precision    recall  f1-score   support

           0       0.65      0.22      0.33        50
           1       1.00      0.26      0.41        50
           2       0.71      0.52      0.60        46
           3       0.92      0.24      0.38        50
```

4	0.58	0.66	0.62	56
5	0.31	0.91	0.46	57
accuracy			0.48	309
macro avg	0.69	0.47	0.47	309
weighted avg	0.68	0.48	0.47	309

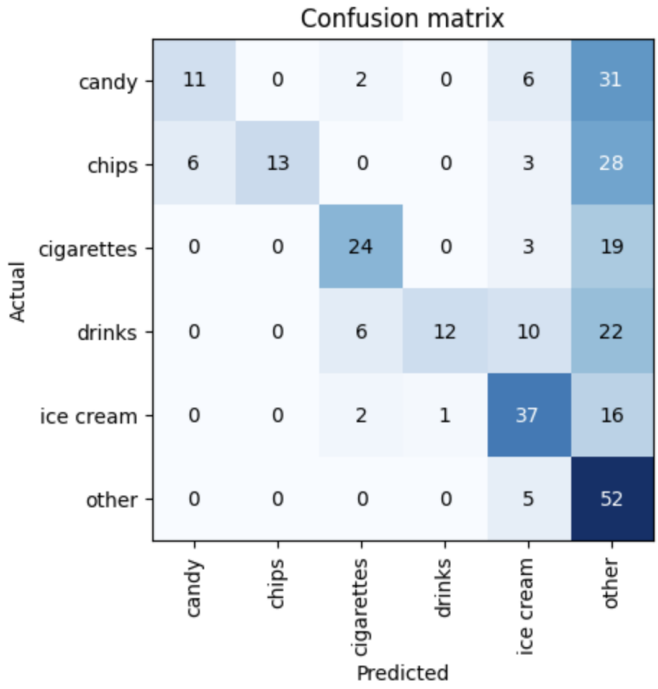


Fig. 11 - Confusion matrix of Resnet on Dataset_1. Note the extremely low precision of 0.31 for "other" category

Candy Other

Fig. 12 - Side-by-side image comparison of "candy" and "other" image. Note the similarities

Although the model has low accuracy of 48%, it actually performs well in some cases; it has high precision scores for chips and drinks categories. The model however had very low precision for the “other” category. This could be for numerous reasons, however, two likely reasons we hypothesize for the high loss is 1) “other” contains multiple categories and groups of items that do not fit in the main categories of candy, chips, drinks, cigarettes, and ice cream; 2) Observe how in Figure 12, the “other” image contains snack bars that are very similar in appearance to chocolate bars in the candy image. The ResNet model was not able to distinguish between these features.

(2.4) Extended Modeling

The next step was to try out other models. ConveNeXt was applied and achieved an accuracy of 0.6084.

Accuracy: 0.6084

	precision	recall	f1-score	support
0	0.83	0.76	0.79	50
1	0.86	0.64	0.74	50
2	0.54	0.98	0.69	46
3	0.87	0.54	0.67	50
4	0.38	0.09	0.14	56
5	0.42	0.72	0.53	57
accuracy			0.61	309
macro avg	0.65	0.62	0.59	309
weighted avg	0.64	0.61	0.58	309

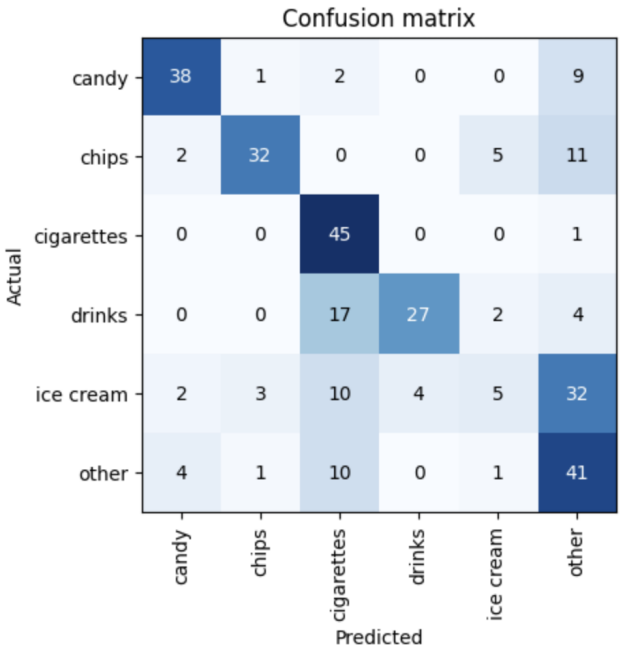


Fig. 13 - Confusion matrix of Resnet on Dataset_1. Note the extremely low precision of 0.31 for "other" category

The model performed relatively well classifying candy, chips, cigarettes, and drinks, except for the 17 False Positives where the model labeled drinks as cigarettes. This is because drinks and cigarettes tend to be uniformly packaged, shaped, and merchandised. The problem categories are ice cream and other. Ice cream had an extremely low recall score of 0.09.

As stated above for the resnet model, "other" is a troublesome category, however in this model it has a decent recall score of 0.72, meaning that the model does a satisfactory job of capturing all of the True Positives for "other". This comes at the cost of precision, where the model's precision for "other" was only 0.42. In other words, the model over-predicted items as "other", especially when it classified ice cream and chips as "other" 32 and 11 times, respectively.

ViT was tested next, with accuracy of 0.8058. A higher level of performance was desired, so the dataset was improved upon; class imbalance was fixed and simple image augmentation was performed. This amounted in a dataset containing 1321 images, and the models were tested again on this new dataset. The overall accuracies of Resnet, ConvNeXt, and ViT on this new and improved dataset were 0.333, 0.8544, and 0.9579, respectively.

(3) Findings

model	accuracy	precision	recall	f-1score
ResNet50	0.33	0.44	0.22	0.29
ConvNeXt	0.85	0.84	0.9	0.86
ViT	0.95	0.94	0.92	0.96

The best model was ViT with an accuracy of 0.95. This is not surprising since ViT is the model with the most sophisticated architecture and uses transformers, while ConvNeXt and ResNet only use convolutions. Of the three models, ResNet actually performed worse on the second dataset. The changes in model performance can be seen in Figure 14, where models tended to perform better on the second dataset.

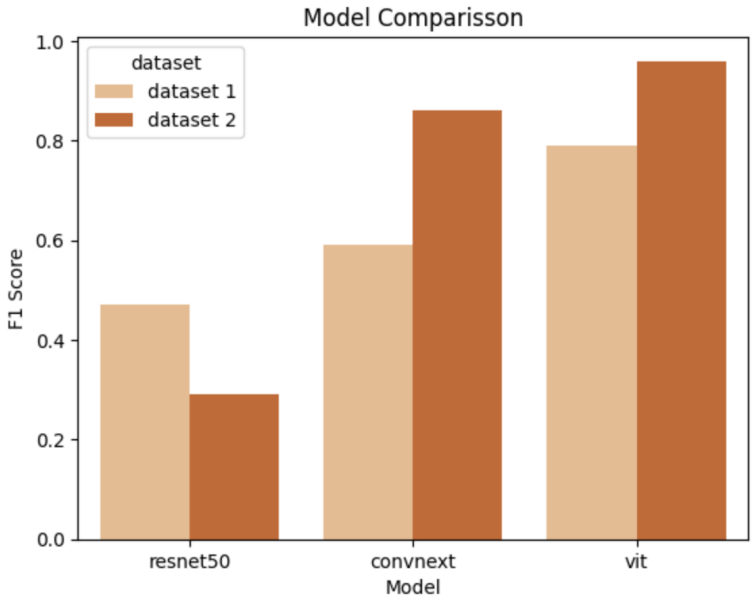


Fig. 14 - F1 scores for models trained on dataset 1 and 2. Models tended to perform better on dataset 2.

We believe the ViT model works the best because it uses transformers under the hood. Transformers were originally trained for sequence to sequence problems in Natural Language Processing. The images presented in this dataset, although they are just images, actually have more in common with language and grammar than they do with images of objects. This is because in merchandising, there is a grammar-like procedure of placing products on a shelf that is similar to the grammar that we use to construct sentences.

In written language, we order sentences according to grammar rules and parts of speech. After years of practice and education, we develop a mental model of how to form our own sentences based on a gut feeling of what sounds right to the ear. In written language, sentences are structured line-by-line, and left to right. We do the same thing when merchandising. Items are placed shelf-by-shelf, left-to-right; potato chips next to potato chips, crackers, and then cookies. You wouldn't want to walk into a store and see sardines merchandied next to chocolate and automotive oil. The ViT model was likely able to learn these key merchandiser features because of it's transformer architecture, while the convolutional models do not have the capacity to characterize these features.

(4) Conclusions and Future Work

This study achieved outstanding results given such few training data and the complexity of the problem. It is remarkable that the model can classify these six categories with an accuracy of 95%. Further work is nevertheless required. One interesting problem to solve would be a segmentation task, where the model is able to segment individual items in each image. Furthermore, it would be nice if there was a model that could detect out-of-stocks. There are many active patents in this area[5].

(5) Recommendations for the Clients

- Deploy the ViT model onto a web-based platform so that merchandisers in the field can utilize the application
- Fine-tune the pretrained model further on proprietary company merchandise photos
- Add more images and categories to the model like Paper Products, Grocery, Health & Beauty Care, etc.
- Test the model on images of merchandise in convenience and retail stores to verify 95% accuracy of model

(6) Consulted Resources

1. Xkcd meme - https://xkcd.com/1425/
2. Residual networks - https://arxiv.org/abs/1512.03385
3. ConvNeXt - https://arxiv.org/abs/2201.03545
4. ViT - https://arxiv.org/abs/2010.11929
5. Patents - https://patents.google.com/?q=(camera+%2b+retail+%2b+machine+learning)&oq=camera+%2b+retail+%2b+machine+learning