PROJECT REPORT

Inventory Monitoring at Distribution Centers

Amazon Bin Image Dataset

Author: Snehal Bhagat
Proposed on Date: 6th June 2022

Programme: Machine Learning Engineer Nanodegree
From: Udacity - School of Artificial Intelligence

1 Definition

1.1 Project Overview

Inventories are the foundation of a successful and functioning e-commerce company. To ensure the continuity of a business and its supply chain, it therefore becomes key to manage its inventory with maximum optimization. Inventory management has a direct impact on the financial performance and growth of an e-commerce business.

Relevant Statistics

1) Warehouse space in the United States costs about $5.08 per square foot[1]
2) American retailers carry about $1.43 in inventory for every $1 of sales[1]

Additionally, such warehouses and distribution centers require constant monitoring to improve their efficiency whether manual or automated. Retail/e-commerce companies frequently struggle with order mix-ups, dead stock issues, insufficient stock levels, and warehouse disorder. In certain cases, inventory maintenance also poses blockers such as staff safety as evidenced by the COVID pandemic, with staff having to work just to maintain a normal supply chain.

Automating inventory management thus becomes key for any major company to optimize their cash flows. In lieu of this, distribution centers use robots to move objects as a part of their operations. Objects are carried in bins which can contain multiple objects. Computer Vision intelligence inbuilt in such robots can help enhance the efficiency of such systems.

To that end, this project aims at addressing one of the tasks in the inventory management lifecycle - that of inventory monitoring and tracking. This is achieved by using Computer Vision techniques to detect the number of objects in a package on the assembly/packaging line. The detected count can then be used to do multiple things, two which are

a) Ensuring the correct number of items are packed and delivered.
b) Automatically tallying the inventory stock.

1.2 Problem Statement

The problem statement for this project is :

“Identify the number of objects in an Amazon delivery package(bin)”

Automatically detecting the number of objects can be used to track inventory and make sure that delivery consignments have the correct number of items

2 Analysis

2.1 Data Exploration

2.1.1 Amazon Bin Images Dataset

The Amazon Bin Image Dataset contains over 500,000 images and metadata from bins of a pod in an operating Amazon Fulfillment Center. The bin images in this dataset are captured as robot units carry pods as part of normal Amazon Fulfillment Center operations[3].

Over 500,000 bin JPEG images and corresponding JSON metadata files describing items in bins in Amazon Fulfillment Centers are available as part of the dataset.

sample image

2.1.2 Dataset Metadata

For each image there is a metadata file containing information about the image like the number of objects, its dimension and the type of object. For this task, we will try to classify the number of objects in each bin.

```
{'B000LRH9J2': {'asin': 'B000LRH9J2',
                'height': {'unit': 'IN', 'value': 2.49999999745},
                'length': {'unit': 'IN', 'value': 5.799999994084},
                'name': 'Mint Leaf Crushed (Castella) 3 oz',
                'normalizedName': 'Mint Leaf Crushed (Castella) 3 oz',
                'quantity': 2,
                'weight': {'unit': 'pounds', 'value': 0.3},
                'width': {'unit': 'IN', 'value': 2.699999997246}},
 'B001005KVS': {'asin': 'B001005KVS',
                'height': {'unit': 'IN', 'value': 4.199999995716},
                'length': {'unit': 'IN', 'value': 4.899999995002001},
                'name': 'Absolute Coatings 3775 Last N Last Wood Finish '
                        'Acrylic Satin Clear, 1 quart',
                'normalizedName': 'Absolute Coatings 3775 Last N Last Wood '
                                  'Finish Acrylic Satin Clear, 1 quart',
                'quantity': 2,
                'weight': {'unit': 'pounds', 'value': 2.3499999980293125},
                'width': {'unit': 'IN', 'value': 4.299999995614}},
 'B00XHYVS16': {'asin': 'B00XHYVS16',
                'height': {'unit': 'IN', 'value': 1.599999998368},
                'length': {'unit': 'IN', 'value': 7.599999992248},
                'name': 'i-smile® 15PCS Replacement Bands with Metal Clasps '
                        'for Fitbit Flex / Wireless Activity Bracelet Sport '
                        'Wristband / Fitbit Flex Bracelet Sport Arm Band (No '
                        'tracker, Replacement Bands Only) & 2PCS Silicon '
                        'Fastener Ring For Free (Set of 15, Large)',
                'normalizedName': 'Set of 15 i-smile® 15PCS Replacement Bands '
                                  'with Metal Clasps for Fitbit Flex / '
                                  'Wireless Activity Bracelet Sport Wristband '
                                  '/ Fitbit Flex Bracelet Sport Arm Band (No '
                                  'tracker, Replacement Bands Only) & 2PCS '
                                  'Silicon Fastener Ring For Free ( , Large)',
                'quantity': 1,
                'weight': {'unit': 'pounds', 'value': 0.39999999966456384},
                'width': {'unit': 'IN', 'value': 5.699999994186001}},
 'B01DP22MF0': {'asin': 'B01DP22MF0',
                'height': {'unit': 'IN', 'value': 1.4173228332},
                'length': {'unit': 'IN', 'value': 2.0472440924},
                'name': 'Anmao 18K White Gold Plated Square Cz Stud Earrings '
                        'Setting For Women Earrings STD-01',
                'normalizedName': 'Anmao 18K White Gold Plated Square Cz Stud '
                                  'Earrings Setting For Women Earrings STD-01',
                'quantity': 1,
                'weight': {'unit': 'pounds', 'value': 0.10999999990775504},
                'width': {'unit': 'IN', 'value': 1.6535433054}}}
```

sample metadata

Each JSON element contains a list of items within an Amazon package. Each item within the package is described by recording the individual item name, dimensions, weight and most importantly **quantity**.

The project will require parsing this file to extract the total quantity of a package mapped against its image location in s3.

An additional input file containing mappings of the image filenames and corresponding number of objects within the bin image is also used. This file will come in handy if training is done in increments of dataset size, thus trying to obtain the highest classification accuracy using the minimum number of images.

A file consisting a mapping of the filenames and corresponding quantities in each image quantity.csv is downloaded to the project folder from the Kaggle link here

```
In [3]:
df = pd.read_csv("../input/amazon-bin-image-dataset-file-list/quantity.csv")
df
```

Out[3]:

	location	quantity
0	s3://aft-vbi-pds/bin-images/00001.jpg	12
1	s3://aft-vbi-pds/bin-images/1.jpg	0
2	s3://aft-vbi-pds/bin-images/00002.jpg	17
3	s3://aft-vbi-pds/bin-images/2.jpg	0
4	s3://aft-vbi-pds/bin-images/00003.jpg	16
...	...	...
536429	s3://aft-vbi-pds/bin-images/535230.jpg	3
536430	s3://aft-vbi-pds/bin-images/535231.jpg	4
536431	s3://aft-vbi-pds/bin-images/535232.jpg	3
536432	s3://aft-vbi-pds/bin-images/535233.jpg	2
536433	s3://aft-vbi-pds/bin-images/535234.jpg	3

536434 rows × 2 columns

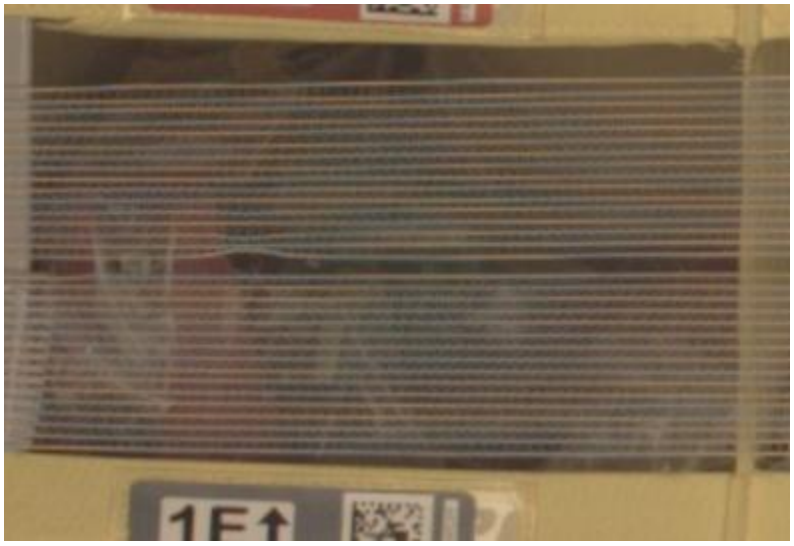
quantity mapping file

2.1.3 Class distributions

The original dataset contains images of bins containing upto 209 objects. While a higher number of images will benefit the accuracy of the model, for this project - only a limited set of bins containing upto 5 objects are being considered.

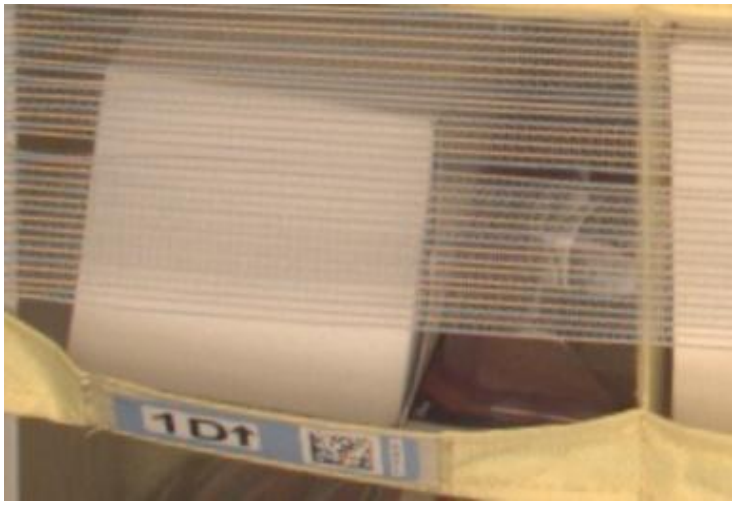
Within each class, a subset of **10000 images** of the total dataset is used. With that the project database consists of 5 classes, each containing **8000 images for training** and **2000 for testing**.

Here are sample images for each class:



CLASS 1

CLASS 2



CLASS 3

CLASS 4

CLASS 5

3 Methodology

3.1 Data Preprocessing and Data Splitting

3.1.1 Convert the data and labels

The mapping file was filtered and only the records containing 1-5 values in the quantity column were used.

```
In [3]: files = pd.read_csv("./quantity.csv")
        print(f"Original dataset size is {files.shape}")

        files = files.loc[(files["quantity"].isin([1,2,3,4,5]))]
        print(f"Dataset size with only classes 1-5 is {files.shape}")
```

```
Original dataset size is (536434, 2)
Dataset size with only classes 1-5 is (352066, 2)
```

Initial filter on the number of images

These records were further filtered such that only the top 10000 records per quantity value were retained.

3.1.2 Split the dataset

The dataset is downloaded in batches per class id using the boto3 s3 client. While downloading 80% of the data is copied to the train folder and the rest 20% to the test folder.

Both Train and Test folders contain subfolders named according to the different classes (viz. 1,2,3,4,5). The number of objects is equal to the name of the folder. For instance, folder 1 has images with 1 object in them.

```
dataset-amazon:
          |
          |
          `----> train:
          |       |-----> 1      #8000 images
          |       |-----> 2      #8000 images
          |       |-----> 3      #8000 images
          |       |-----> 4      #8000 images
          |       |-----> 5      #8000 images
          |
          `----> test
                  |-----> 1      #2000 images
                  |-----> 2      #2000 images
                  |-----> 3      #2000 images
                  |-----> 4      #2000 images
                  |-----> 5      #2000 images
```

folder structure

Finally this data is uploaded to an AWS S3 bucket using the aws cli cp command. The S3 Bucket needs to be created before running the cp command.

3.2 Algorithms and Techniques

The above problem statement can be solved using a Computer Vision Deep Learning Model. Moreover, existing pre-trained models like VGG-16, Resnet-18, Resnet-50, InceptionV3, Xception etc. can be used to reduce training time and costs.

For this project, the Resnet-50 model is chosen as the optimal model as it generates models of size in the range of ~100MBs, owing to actual model weights size which is smaller as it uses global max pooling.[2] This also contributes to its accelerated speed of training.

3.3 Metrics

The evaluation metric used to gauge the performance of the model is Accuracy.

$$Accuracy = \frac{Number\ of\ correctly\ predicted\ images}{Total\ number\ of\ images}$$

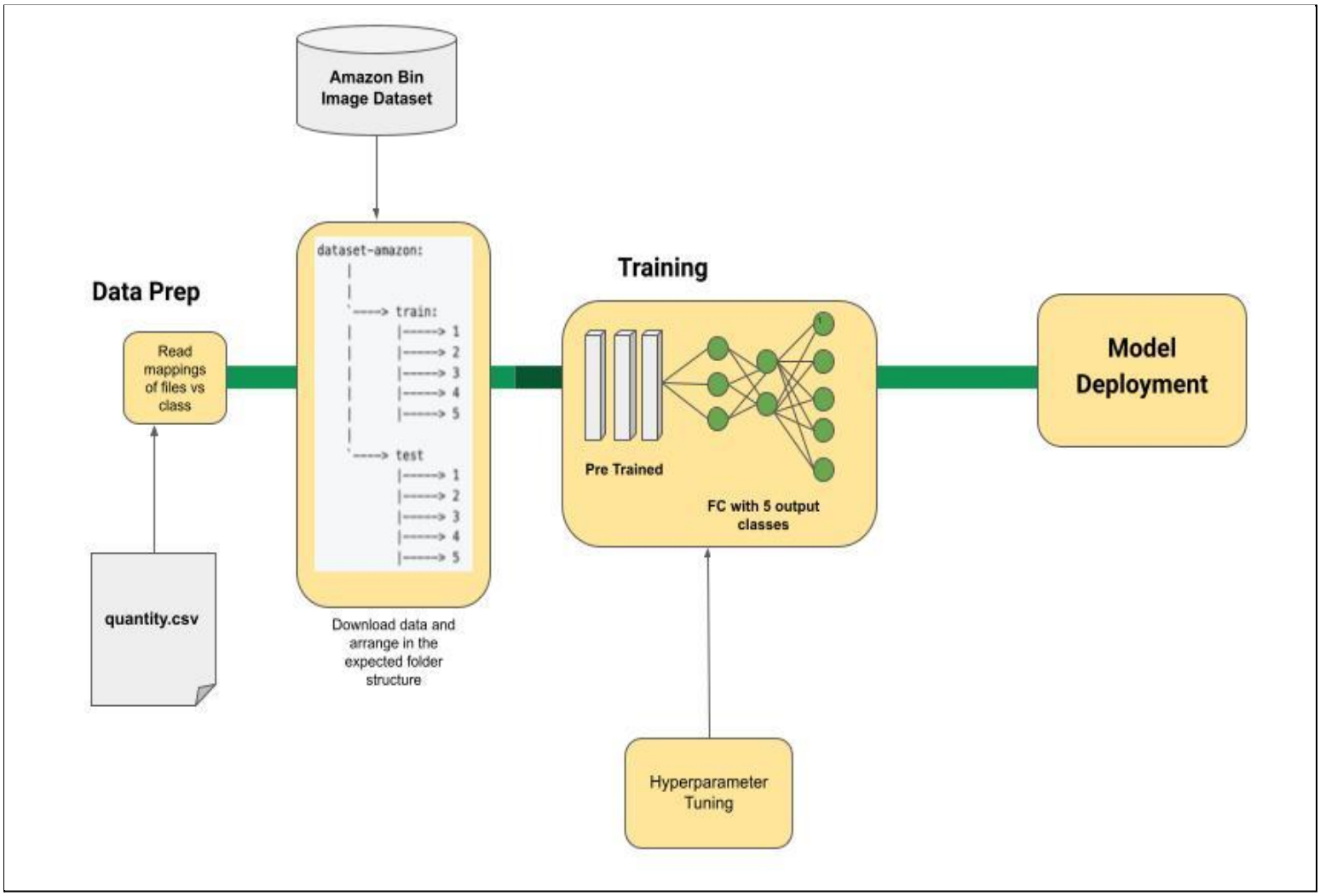
Image created using Online Free LaTex Editor[4]

Since an equal number of images are available within each class of our training dataset, it can be concluded that the dataset is balanced. Accuracy is a useful and meaningful metric when the target class is well balanced.

3.4 Benchmark Model

The selected resnet50 model trained on the given dataset against sample hyperparameters(epochs: 5, learning rate: 0.7) with no data augmentation or preprocessing techniques used will be marked as the benchmark against which further improvements are attempted.

3.5 Implementation



project design

3.5.1 Compiling the model

The fully connected layer added at the end of the Resnet pretrained model was set to output predictions equal to 5 classes.

```
In [1]: import train

In [2]: model = train.net()

Downloading: "https://download.pytorch.org/models/resnet50-19c8e357.pth" to /home/ec2-user/.cache/torch/checkpoints/resnet50-19c8e357.pth

100% 97.8M/97.8M [00:00<00:00, 119MB/s]

In [6]: print(model.fc)

Sequential(
  (0): Linear(in_features=2048, out_features=1024, bias=True)
  (1): ReLU()
  (2): Linear(in_features=1024, out_features=128, bias=True)
  (3): ReLU()
  (4): Linear(in_features=128, out_features=5, bias=True)
)
```

number of output classes

3.5.2 Training & Testing

The training was done on 1 instance of ml.g4dn.2xlarge server. Each training trial of 5 epochs took ~15-20 minutes thus costing ~$0.4

Accelerated Computing	vCPU	Memory	Price per Hour
ml.g4dn.4xlarge	16	64 GiB	$1.505

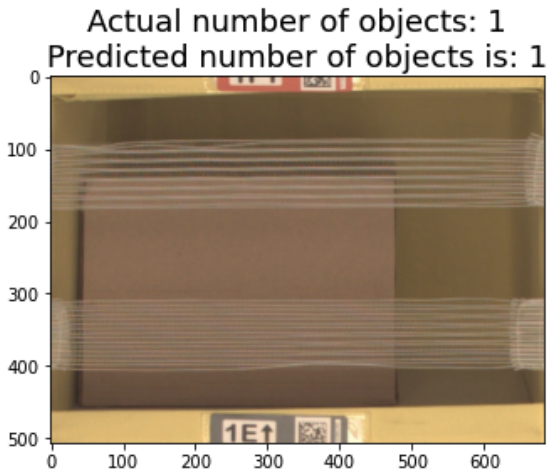
training instance rates

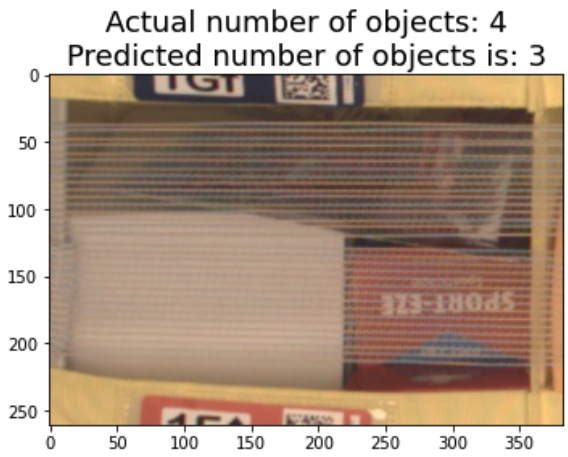
```
[2022-06-04 20:16:45.660 algo-1:26 INFO hook.py:425] Monitoring the collections: losses
[2022-06-04 20:16:45.661 algo-1:26 INFO hook.py:488] Hook is writing from the hook with pid: 26
Epoch 0: Loss 1.5254530243555704,              Accuracy 29.036666666666665%
START TRAINING
Epoch 1: Loss 1.4934473917007447,              Accuracy 31.28%
START TRAINING
Epoch 2: Loss 1.4846882551193237,              Accuracy 31.490000000000002%
START TRAINING
Epoch 3: Loss 1.4802876953125,              Accuracy 32.42333333333333%
START TRAINING
Epoch 4: Loss 1.4785279504776,              Accuracy 32.836666666666666%
START TESTING
Test set: Accuracy: 32% (1582/5000)
```

training and testing epochs

3.5.3 Predictions

Sample predictions on the deployed model





3.6 Refinement

Refinements attempted to optimize the model:

1) Increasing and decreasing the number of epochs

2) Adjusting the learning rate

3) New images added with random horizontal flips

4) New images added with random vertical flips

5) New images added with random rotations of angles ranging between 0 and 180

6) Resize images down to (64, 64) from original (224, 224) --> Reduces the number of features in the model

7) Freeze the layers of the pretrained model to ensure that the unfreezed version is not making the model unnecessarily complex

4 Results

4.1 Model Evaluation and Validation

Here are some of the optimisations attempted to improve the accuracy of the model:

Attempt	Accuracy	Epochs	Learning Rate	No Grad	Image Size Reduction	Data Augmentation
#1	38%	5	0.0775	False	No	No
#2	36% (⇊)	10	0.0775	False	No	No
#3	40% (⇈)	5	0.0775	False	No	Yes
#4	38% (⇊)	7	0.0775	False	No	Yes
#5	43% (⇈)	5	0.0775	True	Yes	Yes
#6	32% (⇊)	5	0.0775	False	Yes	Yes
#7	42% (⇈)	5	0.279	True	Yes	Yes

5 Conclusion

5.1 Reflection

There is no significant change observed by varying the learning rates

The **best model** so far with a 43% accuracy has the following parameters and settings:

1) epochs :"5",
2) lr:"0.07756500209627985"
3) Image size reduced to (64, 64)
4) Data augmented with
   a) Horizontal flips
   b) Vertical flips
   c) Rotations

5) Unfreezing the pre-trained network

Possibly any further improvements to accuracy can only be seen by adding more images to the dataset, as hyperparameter updates modify the accuracy only trivially.

5.2 Further Scope

Future versions of the project could include a **custom deep learning model** trained using an even larger portion of the dataset to account of domain specific intricacies

Citations

[1]https://www.upkeep.com/what-is-inventory-management

[2]https://analyticsindiamag.com/a-comparison-of-4-popular-transfer-learning-models/

[3]https://registry.opendata.aws/amazon-bin-imagery/

[4]https://www.tutorialspoint.com/online_latex_editor.php

References

https://stackoverflow.com/questions/59201907/overfitting-on-image-classification

https://www.kaggle.com/code/williamhyun/amazon-bin-image-dataset-eda

https://www.kaggle.com/code/dhruvildave/starter-amazon-bin-image-dataset