Capstone Project Proposal for Inventory Monitoring at Distribution Centres

1. The Domain and Background of the Project

This Problem has originated from the popular field of Robotics, and its applications in the Industry, like that of Inventory Monitoring and Supply Chain Functions. A lot of Corporations, which handle physical cargo and deal with supply chain of any kind of goods have tried to bring in automation to make these processes more efficient and accurate. A great example of this is Amazon, who is one of the biggest hubs of delivery of all kinds of goods. These goods are often stored in big warehouses. Since the quantity of these items are in a huge amount, to physically do inventory monitoring would require both large and intelligent human resources, which are both expensive and prone to errors.

This is where robots come in to help in Inventory Monitoring. They can be trained with Machine Learning Models, to perform tasks like Object Detection, Outlier & Anomaly Detection and much more. Once trained, these models are scalable, and can be deployed at a low cost for usage in actual warehouses and distribution centres on industry level robots.

2. The Problem Statement

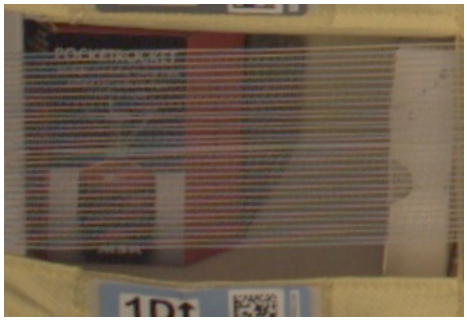
As mentioned above in the domain, distribution centres often have robots which carry objects. These objects are present in bins, and for our problem, each bin can contain 1-5 objects.

The problem we aim to tackle in this project is to count the number of items present in the bin. This is a worthwhile problem to solve, for it has immense real world applications. If we can develop a model, which can take in a picture of a bin, and accurately return the number of objects present in that, we could solve & thus fully automate one crucial step in the Inventory Management process!

3. Dataset and Inputs

The dataset used in this problem is the open source Amazon Bin Image Dataset. This dataset has 500,000 images of bins containing one or more objects present in it. Corresponding to each image, is a metadata file, which contains information about the image, like the number of objects it has, the dimensions and type of objects. For our problem statement, we only need the total count of objects in the image. An example of an image & the corresponding metadata file is shown as below: (Source Reference [1])

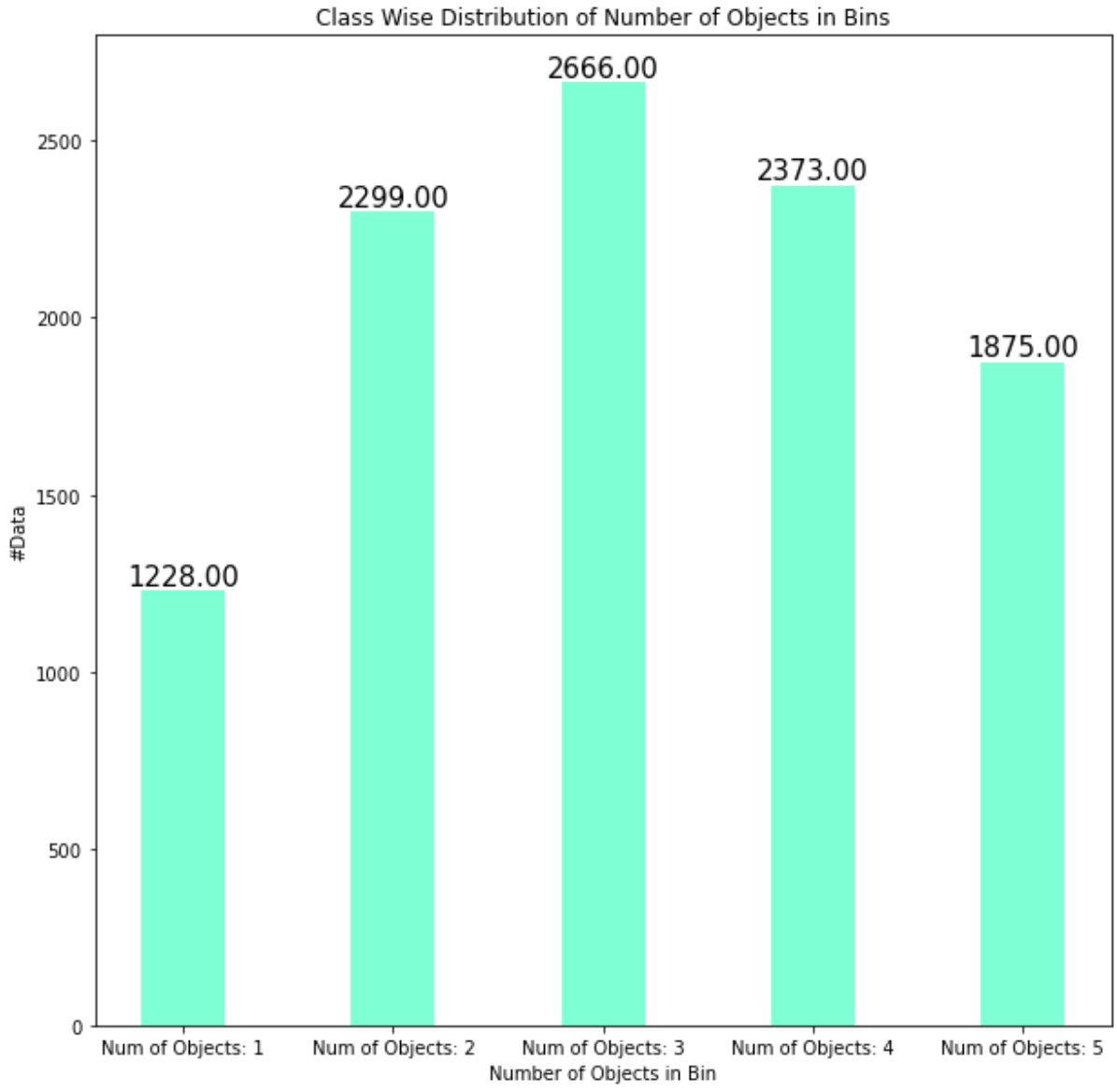

```
{
    "BIN_FCSKU_DATA": {
        "B000A8C5QE": {
            "asin": "B000A8C5QE",
            "height": {
                "unit": "IN",
                "value": 4.200000000000001
            },
            "length": {
                "unit": "IN",
                "value": 4.7
            },
            "name": "MSR PocketRocket Stove",
            "quantity": 1,
            "weight": {
                "unit": "pounds",
                "value": 0.45
            },
            "width": {
                "unit": "IN",
                "value": 4.4
            }
        },
        "B0064LIWVS": {
            "asin": "B0064LIWVS",
            "height": {
                "unit": "IN",
                "value": 1.2
            },
            "length": {
                "unit": "IN",
                "value": 5.799999999999999
            },
            "name": "Applied Nutrition Liquid Collagen Skin Revitalization, 10 Count 3.35 Fl Ounce",
            "quantity": 1,
            "weight": {
                "unit": "pounds",
                "value": 0.3499999999999999
            },
            "width": {
                "unit": "IN",
                "value": 4.7
            }
        }
    },
    "EXPECTED_QUANTITY": 2,
    "image_fname": "523.jpg"
}
```

The “EXPECTED_QUANTITY” field tells us the total number of objects in image. However, since this dataset is too large to use as a learning project, and due to cost limitations on the Udacity AWS Portal, we will be using a subset of the data provided to us by Udacity itself. (Reference [2])

This subset of data has 5 classes, corresponding to number of objects present in the bin: 1, 2, 3, 4 & 5. The total number of images in this subset are: 10,441

The Class Wise Distribution is as follows: (Plotted with code)



We will be further using a split from this for our final testing. If required, we can also get separate data from the AWS Bin Image Dataset, which isn't in our subset, for testing, leaving us with more data for training.

4. Solution Statement

To solve our problem statement, we will use the task of Computer Vision, to come up with a Machine Learning Model, which given an image from our dataset, can identify the number of objects present in it. Essentially, we would be using Multi-Class Image Classification, with Number of Objects from 1-5 as an individual class.

To do this, we can make use of Convolutional Neural Networks, which are a State of the Art technique for Image Recognition tasks. We will leverage Pre-trained models with Transfer Learning to solve the problem.

5. Benchmark Model

In order to find out if the work we did is at par with similar projects, we need a bench mark model or study, we can compare our results with. This benchmark can help guide us to improve our model, and give us an idea if our metrics are making sense, with the constraints we have.

Since there isn't an exactly same model out there, we will first train a CNN Architecture from scratch, and then train another model with Transfer Learning, to compare the results.

Note, due to the constraint of resources available from the Udacity AWS Gateway, the training will not be intense, and the aim of training the two models is for applying the learnings from this Nano Degree. We will be pursuing applying the best practices of MLE we learnt, for this project, rather than achieving state of the art metrics.

6. Evaluation Metrics

To evaluate our model, we need some good metrics, which align with the problem statement. The metrics must be mathematically sound and we should be able to optimise our model for them.

Since we have a Classification Task, we can use Accuracy, Recall, Precision and F1 scores as our metrics. These can be for the overall data, and also class-wise, to identify if a model is doing better on a particular class, or has a high bias for one of them.

Since we have Multi-Class classification, the definition of Precision & Recall must be clearly understood. Let's assume we have 3 classes: A, B & C.

With Multi-Class classification, we have precision and recall for each individual class. For example, metrics for Class A will be defined as follows:

Precision:

#Correctly Predicted Class A Instances / #All Instances predicted as Class A

Recall:

#Correctly Predicted Class A Instances / #Total Class A Instances

F1 Score: 2*Precision*Recall / (Precision + Recall)

So to evaluate our model, we will see per class metrics of Precision, Recall and F1 Score, and see the overall accuracy of the model, which includes all classes, and is defined as:

Accuracy: #Total Correctly Predicted Instances / #Total Instances

Reference [3] for Precision, Recall and F1 for multi-class classification

7. Project Design

The aim of this project is to use SageMaker and good Machine Learning Engineering practices to solve our problem. We define some outline steps, to help accomplish this. **Note, this is not a strict outline and is there to guide the project. The goals could be flexible depending on the outcomes and scope with the resources available.**

I. Load the Corresponding Images from the JSON File and Upload to S3 Bucket, in format for Training Pipeline to load data from there.

   a. In this step we will aim to define the training, validation and test splits for the subset of data we have

   b. We will need to upload them in a S3 Bucket in folder-file format, for the training pipeline to load them with ease

II. Data Visualisation and Data Exploration

   a. In this step we can examine some images class wise, and see if the labels make sense

   b. We can verify that the ratio of train-validation-test split makes sense

III. Ideate Data Pre-Processing and ETL Steps

   a. Since we are working with SageMaker, we will perform the ETL in the training pipeline, but we need to think which transforms are valid for our data distributions

   b. Prepare code for Extract, Transform and Load through Sage Maker

IV. Design Convolutional Neural Network Architecture for our Dataset

a. We need to decide which CNN Architecture is good for us, for both Bench Mark Model and for Transfer Learning

b. Decide which Loss Function and Optimiser we need to use

c. Prepare code for defining our Model and Parameters

V. Train the CNN Model with Sage Maker

a. We need to prepare our code to work with GPUs for faster training

b. We need to finalise the training and testing code

c. We need to decide which instances to use for training, and if we can try Multi-Instance Training

d. We can decide if we want to try for Hyper Parameter Tuning or try parameters from intuition, for refinement

VI. Model Evaluation

a. We need to define the metrics we want to use and code them

b. Evaluate our Model(s) on the test data, with those metrics

c. We can consider deploying the model for inference and using other images to test it

d. Compare Results with Benchmark and Reflect on the result

8. References

[1] https://github.com/awslabs/open-data-docs/tree/main/docs/aft-vbi-pds

[2] https://github.com/udacity/nd009t-capstone-starter/blob/master/starter/file_list.json

[3] https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2