Inventory Monitoring at Distribution Centers

Joan Gerard
28/July/2023

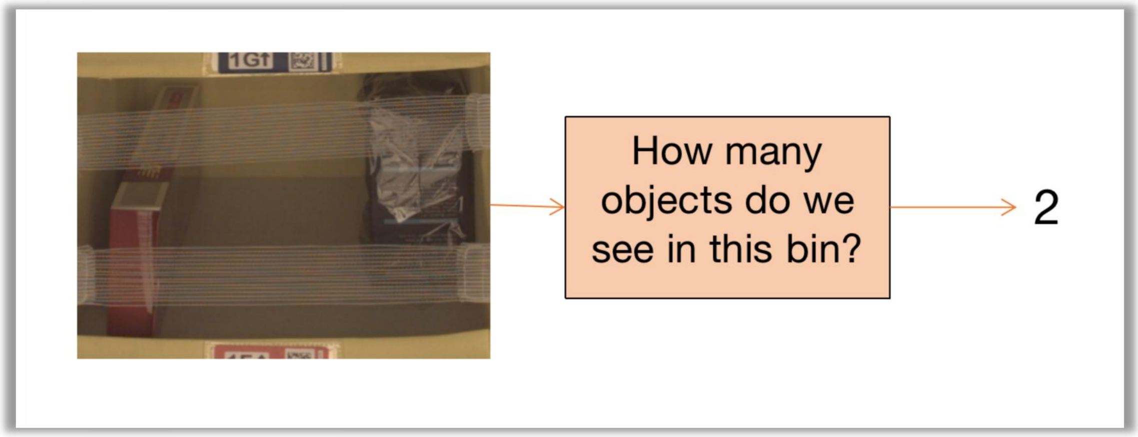
1. Project Overview

Distribution centers often use robots to move objects as a part of their operations. Objects are carried in bins which can contain multiple objects. These objects are then shipped to the customers who order them for instance via Amazon.

Inventory Monitoring at Distribution Centers is an important task for some companies in charge of delivering packages. When you order something from Amazon they need to make sure that they are delivering all the product items you ordered within a single package, nothing more and nothing less. A simple way to verify this is making sure that the number of objects inside the bins matches the number of objects the customer has ordered.

2. Problem Statement

The problem is avoiding distribution centers to deliver packages with missing items. An image of a bin containing the ordered items is provided and the number of objects in the bin needs to be determined.



This task is very easy to accomplish manually when it is only some pictures but what if we generate thousands of them per hour? Here is where the need of producing a model that may help us counting will be helpful so it can process a bunch of images per minute counting the objects in a fraction of milliseconds.

3. Metrics

The metric used for assess the classifier is the accuracy, defined as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \times 100\%$$

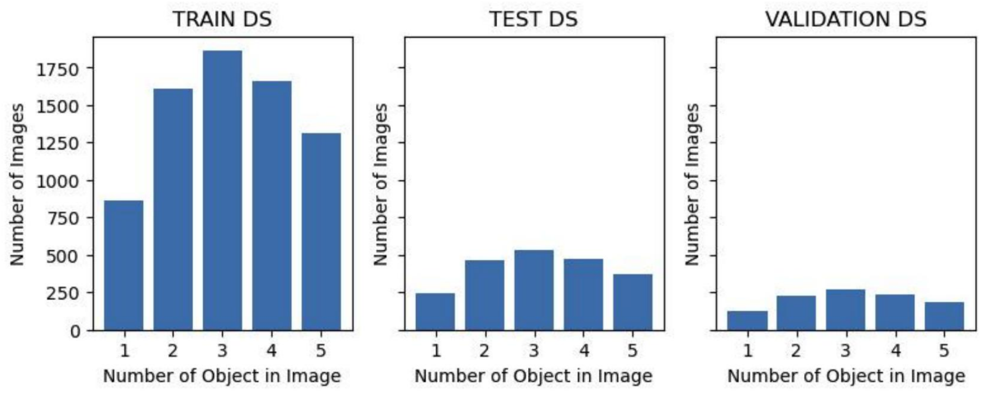
Some previous works see this task as a regression problem instead of a classsification. In that case the use of some other metrics such as *RMSE* are needed; however, this work focuses on the classification part only.

4. Analysis

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. The dataset was obtained from here: https://registry.opendata.aws/amazon-bin-imagery and it contains images of customer's orders before being sent to them. Also contains labeled information about the images such as how many products it has, what the products are, their weights, etc. The dataset is publicly available and it can be accessed either from an S3 bucket or HTTP request.

4.1. Data Exploration

Only a portion of the dataset will be used for this project due to time and budget constrains. A distribution of the data already split in training, testing, and validation datasets is shown here.

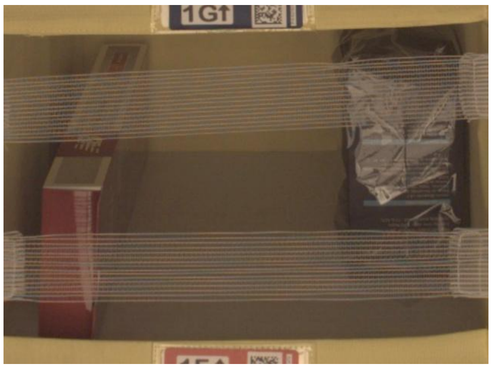


The plan is to work with only 5 classes (number of objects in the bin) and the number of samples per class does not surpases 2000 samples. That will help saving training time but it will unavoidably affect to the final accuracy.

4.2. Exploratory Visualization

The images can be accessed via HTTP or S3 together with metadata information that contains what product items are in the package and how many items there are. This information can be used to generate a dataset with labeled images used for suppervised learning.

The images have different sizes and resolutions with objects of different colors and sizes.



4.3. Algorithms and Techniques

The project will use a pretrained version of ResNet50 which is a Convolutional Neural Network with 50 layers (Convolution + maxPooling). Transfer learning is applied to add some more fully connected layers:

- Dropout of 0.25
- Fully Connected Layer: with 2048 neurons
- Fully Connected Layer: with 128 neurons
- Fully Connected Layer (output layer): with 5 neurons one for each class

The dropout layer cuts off 25% of the connections. This method allows preventing overfitting and improves accuracy. The output layer contains only 5 neurons, each of them predicts one of the classes or number of objects.

4.4. Benchmark

The problem of finding the number of objects given a picture of the product items conained in a bin is not new and many researchers and data scientists tried to resolve it. Park [1] used used the dataset to make object counting, object verification, and object quantity verification training a custom model on top of a ResNet with 34 layers for 40 epochs, a batch size of 128 and obtained an accuracy of 55.67%. Bertorello et al. [2] compared two methods using SVM and different models of CNNs such as ResNet18, ResNet34, and ResNet50 to predict the quantity of the bins obtaining an accuracy of 56.2%

with ResNet34 outperforming SVM by 75%. Xu et al [3] also applied CNNs to solve the problem using data augmentation obtaining an accuracy of 54.64%.

5. Methodology

5.1. Data Preprocessing

The images are downloaded and stored in a structured way, split into training, testing and validation sets and then uploaded to an S3 bucket that can be access for training instances. They are first transformed before training the model, resized to 112x112 pixels and rotated randomly for the training dataset. There is no need to rotate the images for the test and validation dataset but they are as well resized to be 112x112 pixels.

5.2. Implementation

The implementation follows several steps:

- *Data downloading:* the dataset is downloaded locally and then upload to S3 so it can be used by other instances
- *Data cleaning:* some of the images presented incorrect annotations of were very blurry so they were deleted from the dataset
- *Data transformation:* the images were resized to 224x224 pixels and rotated randomly before training the model
- *Model Selection:* hyperparameter tuning was performed with 4 jobs and then the best model was selected to be trained.
- *Model Training:* the best model was trained and evaluated
- *Debug and Profiling:* a complete profiler report was obtained with relevant information such as GPU/CPU utilization and debugger information such as training loss was reported.
- *Model Deployment:* it deploys the best model to a Sagemaker endpoint.
- *Model Inference:* once the model is deployed, it can be used to make predictions for different images never seen before.

All these steps were made in Sagemaker via a Jupyter Notebook report. The tools and resources provided by Sagemaker were fully used such as multi-instance training, hyperparameter tuning, deployment, storing the dataset, etc. Pytorch was the main ML package used for this project.

5.3. Refinement

In order to have s better model some hyperparameters are tuning such as the *learning rate* and *batch size*. The first one may take any value between 0.001 and 0.1 and the second one may take any of these specific values 32, 64, 128, 256, 512. Four jobs were trained using Sagemaker. The script that runs for training during hyperparameters tuning was configured with early stopper which is a method to stop the training if the validation loss does not change in few iterations or starts to increment.

6. Results

6.1 Model Evaluation and Validation

The best model found by the hyperparameter job was the one with a batch size of 128 and 0.00124 of learning rate. The model was trained for 10 epochs and it reports a testing accuracy of 32.41% which is better than a random classifier which were very poor compared to Bertorello et al. [2] who obtained 56% of accuracy.

6.2. Justification

There are some improvements that may be done for increasing model accuracy such as the use of a bigger portion of the dataset and some more transformations to the images such as normalization, data augmentation, etc. Also, more training jobs can be used to expand the searching space for a better configuration of hyperparameters.

Even though the model does not report high accuracy compared to previous works, it makes use of tools for ML in the cloud such as Amazon Sagemaker which is one of the most popular tools in the area. Finally, an end-to-end ML pipeline was created to obtain a model that can predict object counts via an endpoint call.

7. Conclusion

Sagemaker is not an easy to use tool. It is a tool that evolves in time and many things that used to work in the past are going to be deprecated soon. The documentaion is complete enough but the examples provided does not work all the time.

This project presented a lot of difficulties to obtain a final-ready-to-use model because the accuracy of the model presented is low and it may not be sufficient to move it to a production-ready environment. Further work needs to be done to have a better model such as the use of more images from the dataset, more experimentation with some other pretrained models, some more architecture, the use of more hyperparameters, etc. Nevertheless, the present project has accomplished its purpose which is to show how to have a model through the end-to-end pipeline process needed using AWS resources and tools.

8. Bibliography

[1] Park, Eunbyung, Amazon Bin Image Dataset (ABID) Challenge GitHub Repository, https://github.com/silverbottlep/abid_challenge. July, 2017.
[2] Rodriguez Bertorello, Pablo; Sripada, Sravan; and Dendumrongusup, Nutchapol. Amazon Inventory Reconciliation Using AI. December, 2018.
[3] Xu, Zihao and Salloum, Mariam. Deep Neural networks for Object Enumeration. 2018. DOI:10.1109/BigData.2018.8622191
[4] Amazon Bin Image Dataset was accessed on July 2023 from https://registry.opendata.aws/amazon-bin-imagery