AWS Machine Learning Engineer Nanodegree Capstone Proposal - Inventory Monitoring at Distribution Centers

Yue Chang

Feb 26, 2023

Domain Background

The retail industry has been rapidly advancing, with the increasing use of automation and machine learning in various aspects of operations. One such area is the optimization of product storage and retrieval in warehouses, which is critical for efficient order fulfillment. In this context, the Amazon Bin Image Dataset provides a valuable resource for exploring machine learning solutions to object counting and identification within bins.

Object counting in images has been an active area of research, with numerous applications in domains ranging from agriculture to surveillance. Convolutional Neural Networks (CNNs) have shown great success in object detection and counting, and recent research has further advanced these methods with novel network architectures and training techniques. This project aims to build upon this body of work and apply it to the task of counting objects within Amazon bins, with the goal of improving the accuracy and efficiency of product storage and retrieval in warehouse settings.

Problem Statement

The problem to be solved in this project is to accurately count the number of objects in Amazon Bin Image Dataset using machine learning algorithms. The dataset consists of images of bins containing objects of various types and sizes, and the task is to classify the number of objects in each bin. This problem is important because it can help improve inventory management and supply chain operations in warehouses, reducing costs and improving efficiency. The solution to this problem will involve training a machine learning model on the dataset and evaluating its accuracy using appropriate metrics such as accuracy and RMSE. The problem is well defined, measurable, and replicable since the dataset and evaluation metrics are publicly available, and the solution can be reproduced and tested on new data.

Datasets and Inputs

For this project, we will be using the Amazon Bin Image Dataset provided by the Amazon Robotics team, which contains over 535,234 images of items randomly placed in bins. Label distribution are showed as below, and we can see that 90% of bin images contains less then 10 object instances in a bin.

Quantity Distribution
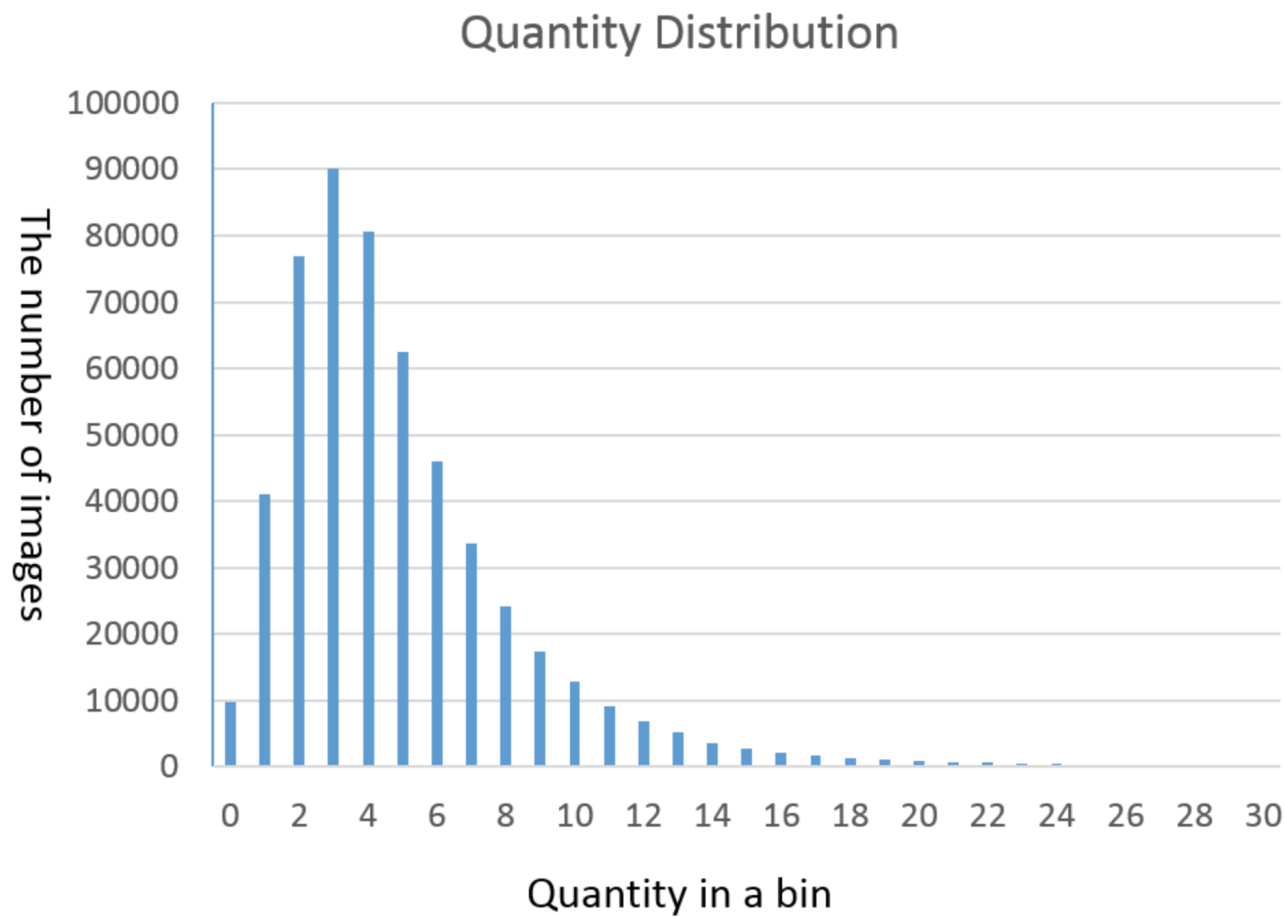
Each image is accompanied by metadata that includes the number of objects present in the image, the dimensions of the objects, and the class labels for each object. Below is an example of metafile and image:
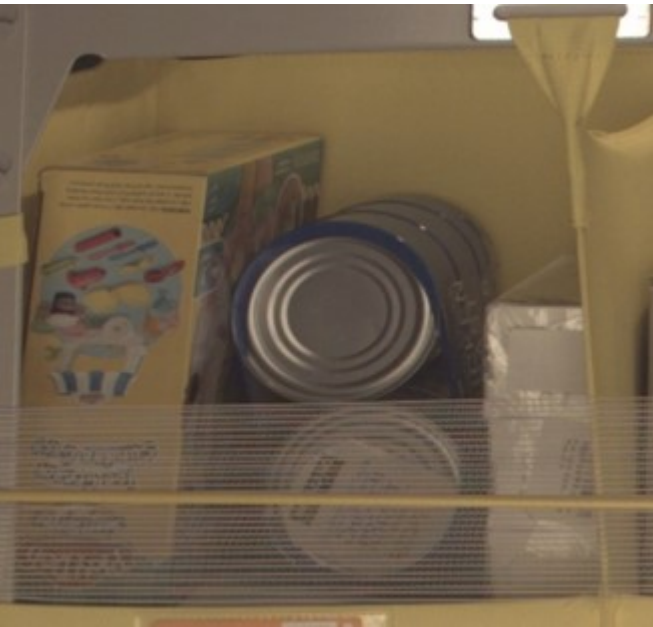

```
{
    "BIN_FCSKU_DATA": {
        "B00CFQWRPS": {
            "asin": "B00CFQWRPS",
            "height": {
                "unit": "IN",
```

```
                "value": 2.399999997552
            },
            "length": {
                "unit": "IN",
                "value": 8.199999991636
            },
            "name": "Fleet Saline Enema, 7.8 Ounce (Pack of 3)",
            "normalizedName": "(Pack of 3) Fleet Saline Enema, 7.8 Ounce",
            "quantity": 1,
            "weight": {
                "unit": "pounds",
                "value": 1.8999999999999997
            },
            "width": {
                "unit": "IN",
                "value": 7.199999992656
            }
        },
        "ZZXI0WUSIB": {
            "asin": "B00T0BUKW8",
            "height": {
                "unit": "IN",
                "value": 3.99999999592
            },
            "length": {
                "unit": "IN",
                "value": 7.899999991942001
            },
            "name": "Kirkland Signature Premium Chunk Chicken Breast Packed
in Water, 12.5 Ounce, 6 Count",
            "normalizedName": "Kirkland Signature Premium Chunk Chicken
Breast Packed in Water, 12.5 Ounce, 6 Count",
            "quantity": 1,
            "weight": {
                "unit": "pounds",
                "value": 5.7
            },
            "width": {
                "unit": "IN",
                "value": 6.49999999337
            }
        },
        "ZZXVVS669V": {
            "asin": "B00C3WXJHY",
            "height": {
                "unit": "IN",
                "value": 4.330708657
            },
            "length": {
                "unit": "IN",
                "value": 11.1417322721
            },
            "name": "Play-Doh Sweet Shoppe Ice Cream Sundae Cart Playset",
            "normalizedName": "Play-Doh Sweet Shoppe Ice Cream Sundae Cart
```

```
  Playset",
              "quantity": 1,
              "weight": {
                  "unit": "pounds",
                  "value": 1.4109440759087915
              },
              "width": {
                  "unit": "IN",
                  "value": 9.448818888
              }
          }
      },
      "EXPECTED_QUANTITY": 3
  }
```

Since our task is object counting, we will be using only the number of objects "EXPECTED_QUANTITY" present in each image as our ground truth label. And on the dataset challenge page here, useful codes for dataset preparation and development kits has been provided, we will be using these codes for our project.

We will be using Amazon S3 to store and manage our dataset, which will allow us to easily access the data from within Amazon SageMaker. Additionally, the dataset is into two levels of difficulty(moderate and hard). We will be train and test over the moderate bin images that contain upto 5 objects.

References:

ABID Challenge Dataset. (n.d.). GitHub. https://github.com/silverbottlep/abid_challenge.

Solution Statement

The solution to this problem involves training a deep learning model using the ResNet-34 or ResNet-50 architecture to predict the object counts in an image from 0-5. The ResNet-34 or ResNet-50 model will be fine-tuned on the given dataset using transfer learning with pre-trained Torch models. The model will be trained using SageMaker and hyperparameters will be tuned using SageMaker's hyperparameter tuning functionality. The model will be evaluated on the validation set using metrics such as accuracy, RMSE. The final model will be deployed and tested on a test set to evaluate its performance. This solution can be easily reproduced using the same dataset and models, and can be extended to other similar object counting problems.

Benchmark Model

For the benchmark model, we will use a deep convolutional classification network for counting tasks, details here . The model uses ResNet-34 architecture and is trained from scratch on the dataset, where each image is classified into one of six categories (0-5). The metadata format for both the training and validation set is a list of [image idx, count] pairs. The training script runs for 40 epochs with a batch size of 128, and the learning rate is decayed by a factor of 0.1 every 10 epochs. The best validation accuracy of 55.67% and an RMSE of 0.930 was achieved at 21 epochs, after which the model starts to overfit.

Benchmark model can be downloaded and used for evaluation. And this model's accuracy, RMSE, per-class accuracy and RMSE are also provided. These metrics will be used to compare the performance of our proposed solution.

Evaluation Metrics

For this object counting problem, we can use accuracy and root mean square error (RMSE) as evaluation metrics.

Accuracy measures the proportion of correctly predicted object counts to the total number of object counts in the validation set. It is computed as:

```
accuracy = correct predictions / total predictions
```

where `correct predictions` refer to the number of images where the predicted count matches the true count, and `total predictions` refer to the total number of images in the validation set.

RMSE measures the average difference between the predicted and true counts. It is computed as:

```
RMSE = sqrt(sum((predicted_count - true_count)^2) / n)
```

where `predicted_count` and `true_count` are the predicted and true counts for each image, respectively, and `n` is the total number of images in the validation set.

Project Design

The proposed solution for the Amazon Bin Image Dataset (ABID) is to develop an image-based object counting model using a pre-trained convolutional neural network (CNN). The following is a high-level workflow that outlines the steps involved in this approach:

1. Dataset Preparation

   The first step of the project is to prepare the dataset for training and testing the model. The ABID dataset consists of millions of images that are categorized into different classes. For this project, we will only use the moderate data which has classes from 0-5. This data will be extracted from the ABID metadata.

   The extracted data will then be prepared for model training by resizing the images and generating ground truth (gt) labels for the object counting task. The image resizing and gt label generation will be performed using the tool provided in the ABID challenge page. The tool also includes a train-test split, which will be used to separate the data into training and testing sets.

   To automate this process, AWS ETL (Extract, Transform, Load) tool in SageMaker - the processing job will be used. This tool is useful for cleaning and transforming data from various sources and can help with the data preparation process.

2. Hyperparameter Tuning

   After the data has been prepared, the next step is to tune the hyperparameters of the CNN to achieve the best possible performance. For this task, Amazon SageMaker's Hyperparameter Tuning service will be utilized. Hyperparameter tuning is a process of finding the best set of hyperparameters that will result in the highest model accuracy.

   SageMaker's Hyperparameter Tuning service automates the hyperparameter tuning process by running multiple training jobs in parallel, testing different combinations of hyperparameters. The best combination of hyperparameters will be chosen based on the highest validation accuracy.

3. Model Training and Evaluation

   Once the best hyperparameters have been determined, the next step is to train the CNN on the prepared dataset using the chosen hyperparameters. The CNN architecture used in this project will be based on a pre-trained model, such as ResNet50, that has been trained on the ImageNet dataset. This will allow us to leverage the pre-trained model's feature extraction capabilities and fine-tune the last few layers to improve performance on our specific task.

   The trained model will then be evaluated using two metrics: accuracy and root mean squared error (RMSE). Accuracy is a standard classification metric that measures the percentage of correctly classified images. RMSE, on the other hand, is a regression metric that measures the difference between the predicted and ground truth counts. Both metrics are important for evaluating the performance of an object counting model.

   To ensure that the proposed model outperforms the benchmark model, we will compare the model's accuracy and RMSE scores with the benchmark model. The benchmark model is a simple ResNet-34 model that trained from scratch.

4. Deployment

   Finally, after the proposed model has been trained and evaluated, it will be deployed to an Amazon SageMaker endpoint to allow for easy inference on new, unseen data. The endpoint can be integrated with other AWS services, such as AWS Lambda, to enable automated and scalable model predictions.

In conclusion, the proposed solution for the Amazon Bin Image Dataset involves preparing the data, tuning hyperparameters, transfer learning, training and evaluating the model using accuracy and RMSE, and finally deploying the model to an AWS endpoint. By following this workflow, we aim to develop an accurate and scalable object counting model that can be used for a variety of applications.

```
                +---------------+
                | Extract data  |
                +---------------+
                        |
                        |
                +---------------+
                | Prepare data  |
                +---------------+
                        |
                        |
                +----------------+
                | Preprocess data |
                +----------------+
                        |
                        |
                +---------------------+
                | Tune Hyperparameters |
                +---------------------+
                        |
                        |
                +-------------+
```

```
            | Train Model |
            +-------------+
                   |
                   |
            +---------------+
            | Evaluate Model |
            +---------------+
                   |
                   |
            +--------------+
            | Deploy Model |
            +--------------+
```