AWS MACHINE LEARNING ENGINEER NANODEGREE

CAPSTONE PROJECT

INVENTORY MONITORING AT DISTRIBUTION CENTERS

BY

OGUNDIPE, ABDULLATEEF

JANUARY 25, 2022

Contents

DEFINITION

Project Overview

The project aims to build a model that can accurately classify the number of objects in a bin. This system uses the power of deep learning to track inventory and ensure delivery consignment has the correct number of items. In extensive retail facilities such as Walmart and Amazon, inventory and monitoring are very labour intensive, involve high risk, require multiple people, and are prone to error. The process will improve significantly by automation whereby a vision-based method using deep learning to count the number of items in a bin before delivery

Problem Statement

The distribution centres need to ensure the correct number of items placed in the consignments for delivery. Since robots are used as part of their operations to move objects, a model needs to be integrated into the robot to count things correctly as it forces them to the bin. This system tracks inventory and ensures consignments have the correct number of items.

Metrics

The evaluation metric used for this task is the cross-entropy loss function. Cross-Entropy is a good loss function for Classification Problems because it minimizes the distance between two probability distributions - predicted and actual.

ANALYSIS

Data Exploration

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 capture robot units carrying pods as part of normal Amazon Fulfillment Center operations. A metadata file contains information for each photo, like the number of objects, their dimension, and the type of object.

The total number of images is 535,234, with an average quantity in a bin of 5.1 and several object categories of459,476. Typical images in the dataset have a bin containing multiple object categories and various instances. The corresponding metadata exist for each bin image, including the object category identification (Amazon Standard Identification Number, ASIN), quantity, size of objects, weights, and so on. The size of bins is dependent on the size of things in them. The tapes in front of the bins prevent the items from falling out of the bins, and sometimes it might make the objects unclear.

A typical example of a bin image and its metadata file is shown below;

{

```
  "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
}
```


The data above is an image(jpg) and metadata(JSON) pair. This image contains three different object categories. For each category, there is one instance. So, "EXPECTED_QUANTITY" is 3, and for each object category, the "quantity" field was 1. Unique identifier("ASIN") is assigned to each object category, e.g., here "B00CFQWRPS", "B00T0BUKW8", and "B00C3WXJHY".

The dataset to be used is the subset provided by Udacity

Images are located in the bin-images directory, and metadata for each photo is located in the metadata directory. Images and their associated metadata share simple numerical unique identifiers.

- How are the images formatted?

  There are two sets of inputs for the model training
  1. Images for the model, which is available in the source as a JPEG file
  2. JSON format with metadata for the image

- Are the dimensions consistent?

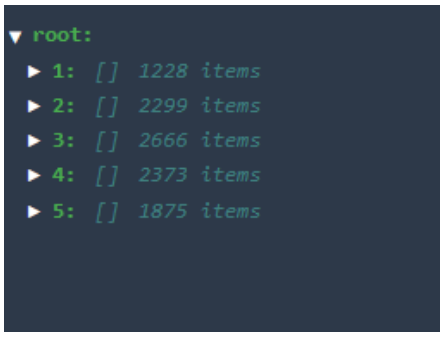
  The dimension of the images is consistent

- Are there colour layers?

  There are colour layers in the images

- How are the dataset classes balanced?

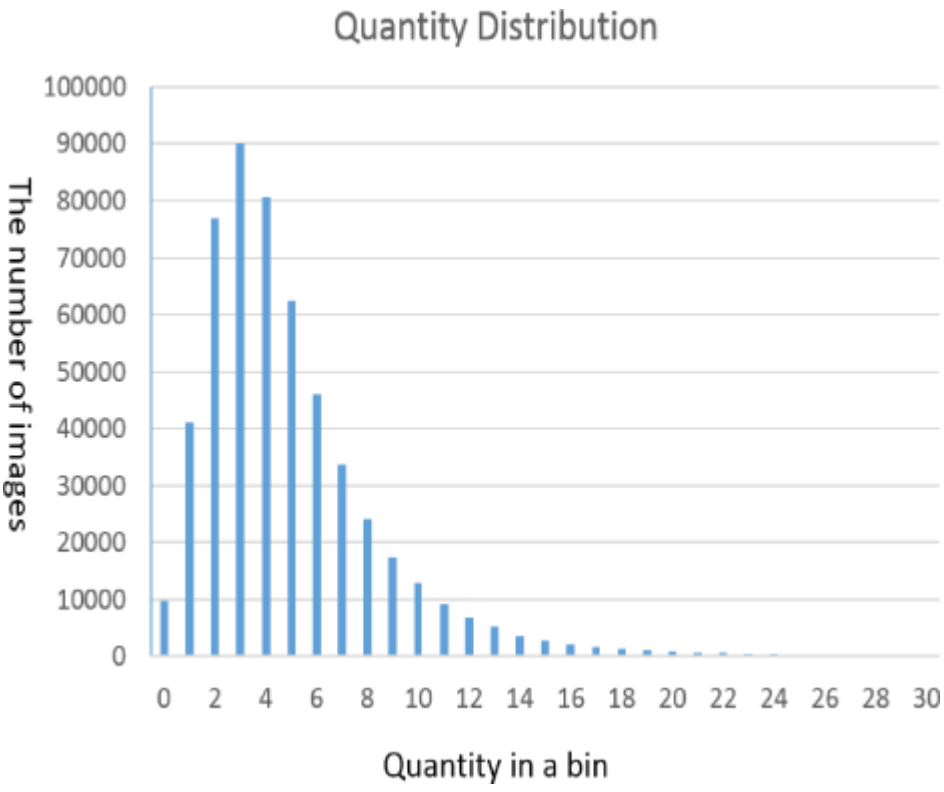
The dataset classes are not balanced. The picture below shows the count of the dataset for each class



Data available at: https://registry.opendata.aws/amazon-bin-imagery/

Exploratory Visualization

The plot below shows the distribution of the number of products in a bin. Here we can see 90% of bin images contain less than ten object instances in a bin



Algorithm and Techniques

The algorithm used as part of the solution is based on Convolution Neural Network architecture. The images are fed into the network to train a model. The model was used for final classification based on the count of the goods in the picture.

• PyTorch Deep Learning Framework

• AWS Sagemaker instance will be initiated, and the image stored in the S3 bucket will be used as part of the data pipeline.

• Model will be tuned, and model with the best hyperparameter will be identified and deployed

A Convolutional Neural Network is trained using transfer learning for this image classification task. Here a pretrained model called '**resnet50**'has been used to finetune on the amazon bin image dataset.

To finetune the resnet50 CNN,

The following steps are performed:

• create or load the pre-trained model

• freeze all the convolutional layers

• add a fully connected layer and train it.

In this training paradigm, the forward pass will run for the whole network, and since the convolutional layers are frozen, the backward pass (and update weights) will run for only the fully connected layers.

The following hyperparameters are used during tuning and training;

• training length (number of epochs)

• batch size (number of images to look at once during a single training step)

• learning rate (how fast to learn)

As this is a multi-class classification problem, the Cross-entropy loss function is used as the cost function, and Adaptive Moment Estimation (Adam) method is used as an optimizer.

Benchmark

The benchmark for this project is to be able to get the model accuracy of **55.67%** or more

METHODOLOGY

Data Preprocessing

Since this is a large dataset, only a subset of the data has been used for this project. The subgroup of the bin images dataset was chosen so that the number of objects in a bin range from '1' to '5' only.

The subset of data created is arranged in subfolders. Each of these subfolders contains images where the number of objects is equal to the folder's name. For instance, all images in folder 3 have ideas with three things. The number of objects will serve as the categories for classifying bin images.

The list of file names to be downloaded is collected in file_list.json. The metadata(JSON) filenames are arranged in JSON objects using the number of images in the bin as the 'key'.

The number of images in each class is shown in the table below

Category (Number of objects in a bin)	**Total Images**
1	1228
2	2299
3	2666
4	2373
5	1875

Implementation

Platforms

The AWS cloud platform was used for model training and deployment in this project. The resources on this platform include; AWS Sagemaker: This is a dedicated service for machine

learning model and training purposes. The model was used to train the deep learning model for my capstone project. Also, s3 was used for data storage and storing training models artefacts. The model will be deployed using Endpoint services.

Setup AWS SageMaker Notebook instance

Create and open a Sagemaker instance, and I have chosen the 'ml.t2.medium' instance type. This instance comes with 2vCPUs and 4GiB memory. This configuration is good enough for running the project jupyter notebook, and the price is also one of the least.

Then create or load the project files required to your workspace.

Training Script (train.py / hpo.py / inference.py)

The training script takes the hyperparameters as arguments and reads, loads and preprocesses the train, test and validation data. Then after creating and loading the pretrained 'resnet50' model, it trains the model using the given parameters. hpo.py: used for hyperparameter tuning job. train.py: used for finetuning the model with the best hyperparameters. inference.py: used for deploying the trained model.

Submission Script (sagemaker.ipynb)

This jupyter notebook has the code cells to interface with the sagemaker and submits all the jobs. The following positions are done using this script;

- Data download and preparation: Download the Amazon bin image dataset[1]. The file_list.json has the list of files to download. The dataset is splitted into train, test and validation directories in the ratio of 8:1:1.
- Data Upload to S3: Upload the downloaded data to the required S3 bucket.
- Setup the Hyperparameter search space
- Design the model debugging and profiling rules
- Create the model estimator
- Create the Hyperparameter tuning job
- Submit the tuning job

- Fetch the best training job and the hyperparameters
- Finetune the model with the best hyperparameters
- Deploy the model
- Query the model with a bin image and get the prediction

Refinement

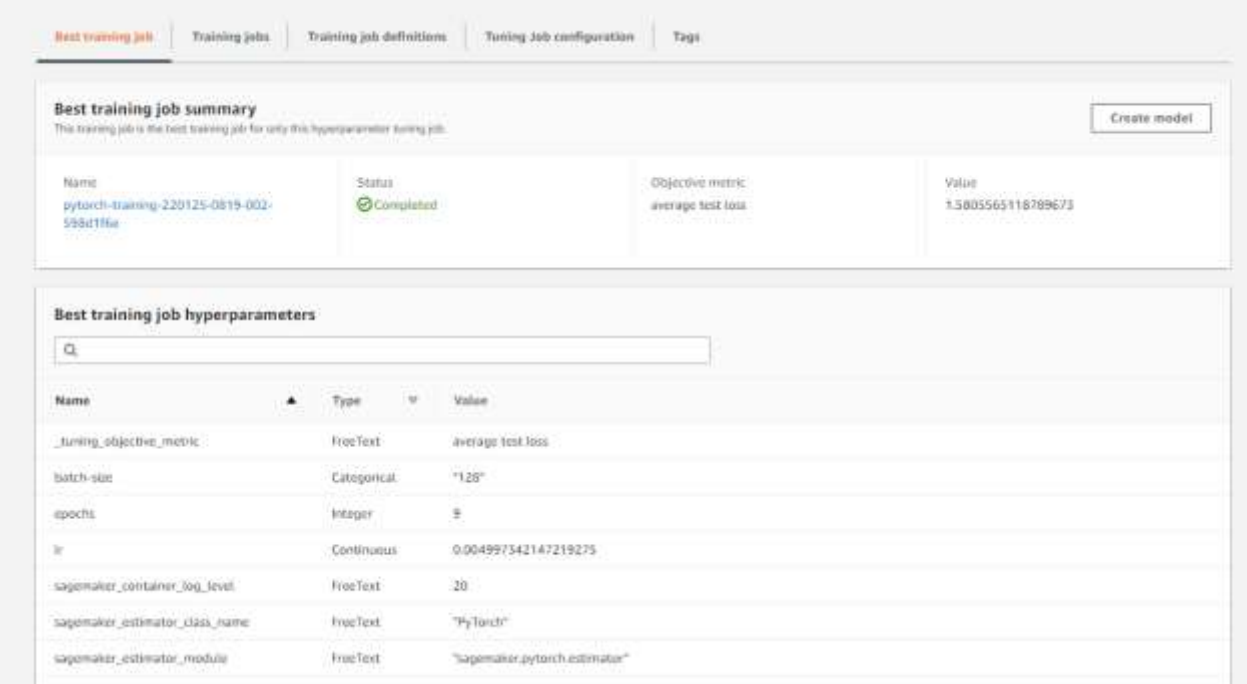
In this project, I created a deep learning workflow using AWS SageMaker. The deep learning model trained the image classification model on the Amazon bin image dataset. The training process can be performed iteratively with various parameters and larger datasets to get a better performing model with minimal script changes.

Initially, I used the pretrained model '**resnet50**' to finetune the model on the data subset using fixed hyperparameters; The model test set loss and accuracy are;

Test set: Average loss: 1.5805392183106521

Testing Accuracy: 0.25574710965156555

I performed hyperparameter tuning with the subset of the bin images dataset. Below is the screenshot of the hyperparameter tuning job with the best training job hyperparameters;



The best performing hyperparameters are;

{'test-batch-size': '100',

'epochs': '9',

'batch-size': 128,

'lr': '0.004997342147219275'}

RESULTS

Model Evaluation and Validation

The final model has been deployed with the best performing hyperparameters. I used PyTorch Model for deploying the model.

Once the model is successfully deployed on Endpoint, I queried using the predict () method.

Screenshot of the endpoint results;

```
In [60]: from PIL import Image
         import io
         Image.open(io.BytesIO(img_bytes))
```

Out[60]:


```
In [61]: response = predictor.predict(img_bytes, initial_args={"ContentType": "image/jpeg"})

In [62]: type(response)
Out[62]: list

In [63]: response
Out[63]: [[-0.1935795694589615,
           0.01356017217040062,
           0.010463185608386993,
           0.06341414898633957,
           -0.06604141741991043]]

In [64]: import numpy as np
         np.argmax(response, 1)
Out[64]: array([3])
```

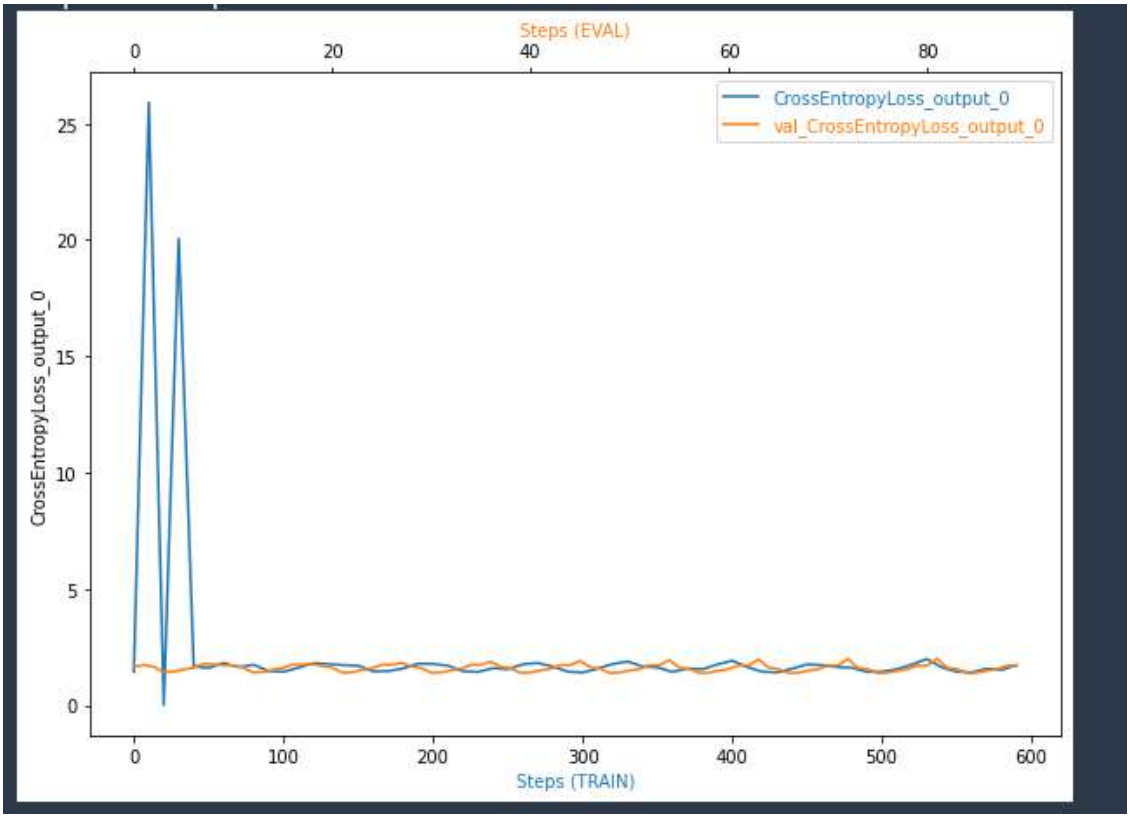
Model Performance and Result

The overall workflow of the model training looks robust. However, the model needs more training with larger datasets and exploring iteratively for better performing CNN models and hyperparameters.

An important thing to consider is using a larger dataset and performing more preprocessing on the dataset

Justification

- Model performance shows a steady decrease in the cross-entropy loss, which sustains over the training steps.
- Validation loss is also consistent with this observation



CONCLUSIONS

The AWS SageMaker provides a robust platform for developing machine learning models. There are debugger and profiling libraries or services that can be useful in debugging and analyzing the model training process.

To build an ML model, some of the essential things to take care of include but are not limited to are accuracy and quantity of the dataset, neural network model architecture, different parameters, the loss function, optimizer functions and cost of the resources.

Also, I think things like increasing the training time, using more extensive datasets and model architectures with a more significant number of layers blindly may not work, and one has to find the optimum values or resources when building an ML model

RESOURCES AND REFERENCES:

1. Amazon Bin Image Dataset : https://registry.opendata.aws/amazon-bin-imagery/
2. https://github.com/knareshkumar/InventoryMonitoringAtDistributionCenters
3. https://www.researchgate.net/publication/342529859_Inventory_management_using_Machine_Learning
4. Documentaions : https://github.com/awslabs/open-data-docs/tree/main/docs/aft-vbi-pds
5. Amazon Bin Image Dataset(ABID) Challenge : https://github.com/silverbottlep/abid_challenge
6. Project Starter files : https://github.com/udacity/nd009t-capstone-starter
7. https://ieeexplore.ieee.org/document/9268394
8. https://github.com/Srimeenakshi20/Inventory-Monitoring-at-Distribution-Centers