Machine Learning Engineer Nanodegree

AWS Capstone Project

Ledezma, Lucia del Valle

April, 2023

Table of Contents

1. Introduction

Computer vision is a field of artificial intelligence that enables computers to derive meaningful information from digital images, videos and any other visual inputs [1]. For this project the subfield related to is Image Classification, it is the task of identifying images and categorizing them in one of several predefined distinct classes [2]. A convolutional neural network is a method of deep learning that takes an input image and assigns importance (learnable biases and weights) to various objects in the image, distinguishing one from the other [3]; furthermore, it has demonstrated exceptional image classification, recognition, segmentation, and retrieval [4].

2. Problem statement:

In a distribution center, all the objects are managed by robots, these robots are responsible to take every object into bins. How can we count how many objects there are in each bin?. The objects can be countered by a system in order to get an inventory. This involves knowing how to classify the images into object's amount labels by using Computer Vision techniques.

The images look like:

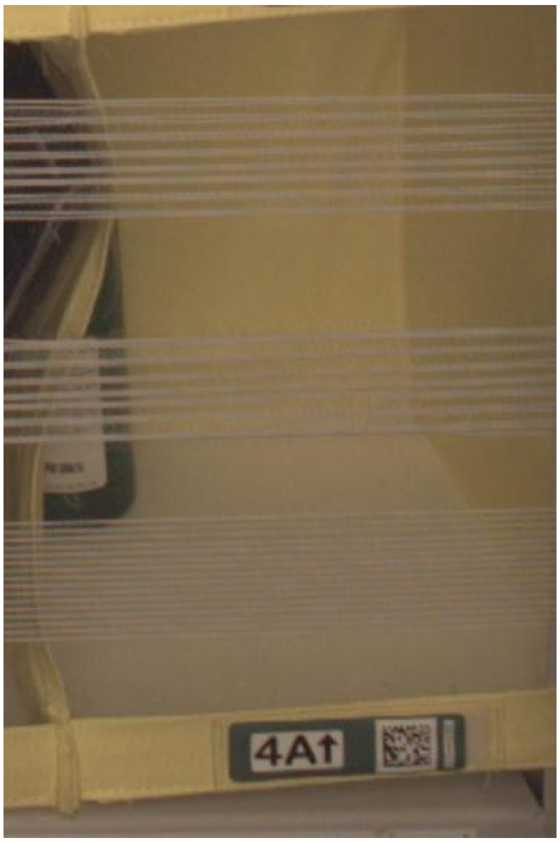

Fig 1. Image of a bin with only one object

3. metrics:

Accuracy is one of the most important evaluation metrics, this represents the percentage of correct predictions of the model:

$Accuracy = (Correct\ Predictions)/(\ total\ predictions)$

By having the confusion matrix, it could being explained as:

$Accuracy = (TP + TN)/(TP + TN + FP + FN)$

4. Data

Amazon Bin Images is the dataset for this project, I only used a subset of this. I downloaded it from s3://aft-vbi-pds/.

Data was provided into 5 folders, each folder is a label class (number of objects in a bin), in other words, each image is classified into the folder according to the number of objects it contains.

It was necessary to split the data into the folders train, test and validation, so the model could be trained by using the ImageFolder method of Pytorch and create the data loaders.

To accomplish this objective two methods were developed : train_test_split which does the splitting, and write_sample_elements for saving the images into its respective folder.

Finally, the splitted dataset folders were uploaded to s3, so the model can train with it.

4.1 Exploration

To detail the dataset, is necessary to understand their distribution:

`

1. For the training dataset, 70% of the original data were used.
2. For the test set, 20% of the original data
3. Finally, for the validation set, only 10% of the original data.

Table 1. Original Dataset

Original Dataset	
Labels	Number of objects
1	1228
2	2299
3	2666
4	2373
5	1875

As we can see, We have different amounts of images in each class.

Table 2. Train Dataset

Training Dataset	
Labels	Number of objects
1	863
2	1613
3	1869
4	1664
5	1315

Table 3. Test Dataset

Test Dataset	
Labels	Number of objects
1	244
2	458
3	532
4	473
5	374

Table 4. Validation Dataset

Validation Dataset	
Labels	Number of objects
1	121
2	228
3	265
4	236
5	186

4.1.2 Visualization

We can observe the datasets distributions with the plot bars below:

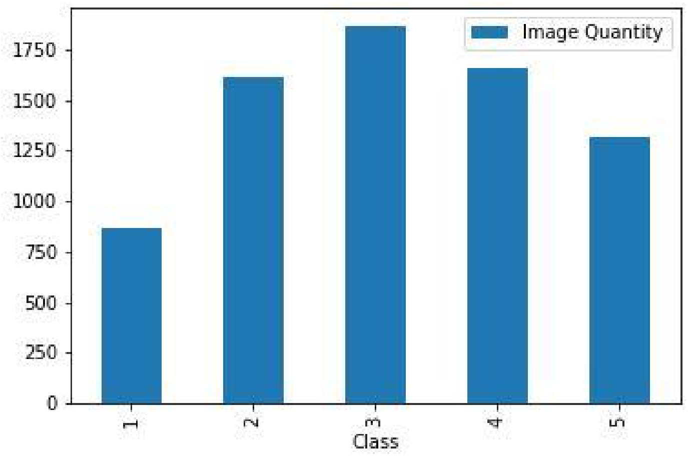

fig. 2 Train dataset bar plot.

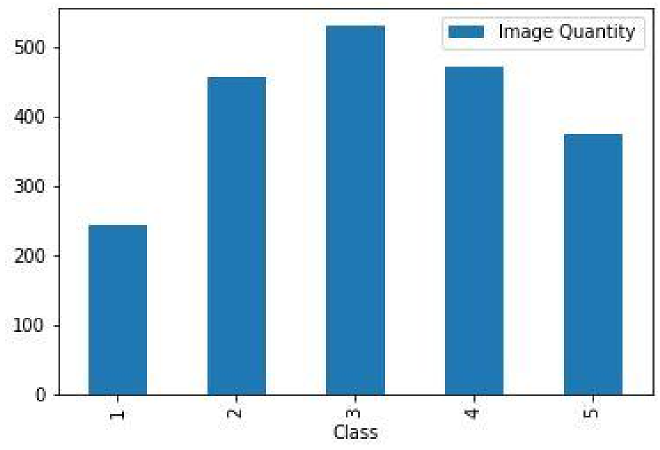

Fig 3. Test dataset bar plot.

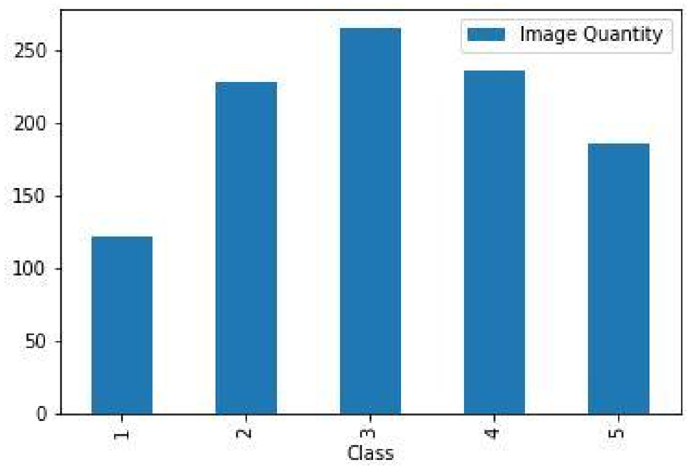

Fig 4. Validation dataset bar plot.

5. Model training

For this step I am going to train two deep convolutional neural networks. Deeper neural networks are more difficult to train. With Resnet, it becomes possible to surpass the difficulties of training very deep neural networks [5].

All the steps below are executed by using Sagemaker environment, we can access all scripts in the github repository of this capstone project.

5.1 Benchmark Model

ResNet18

ResNet-18 is a convolutional neural network that is 18 layers deep. We can load a pre-trained version of the network trained on more than a million images from the ImageNet database. The pretrained network can classify images into 1000 object categories, such as keyboard, mouse, pencil, and many animals. As a result, the network has learned rich feature representations for a wide range of images. The network has an image input size of 224-by-224 [6].

This is the architecture schema:

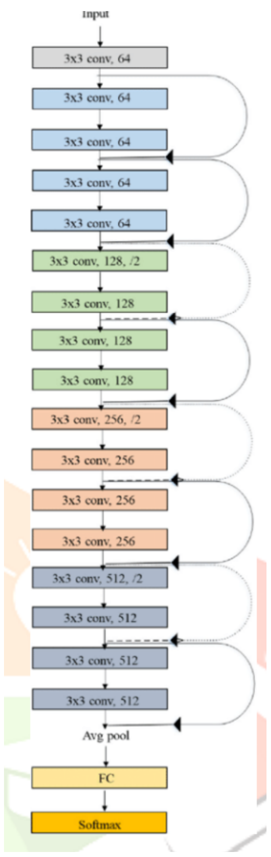

Fig 5. resnet18 [7]

I used the resnet18 pre-trained model as a base for creating my benchmark model, I did transfer learning adding the output layer 5 classes according to the number of objects in each bin.
In order to accomplish this objective, I trained the model by executing the same script of the proposal model, setting the model name parameter ‘model-arch’ with ‘resnet18’. The training process was done with the same dataset explained before.
I trained the model by setting:
hyperparameters = {'batch-size': 64, 'lr': 0.009, 'model-arch': 'resnet50', 'model-n-classes': 5, 'epochs': 6}

5.2 Proposal Model

ResNet50

ResNet-50 is a convolutional neural network that is 50 layers deep. This model is an improvement of the previous one.

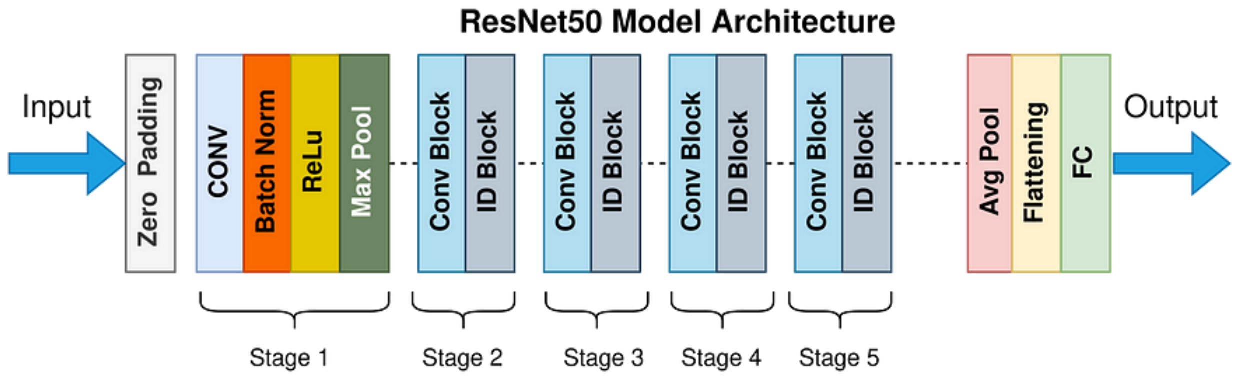

Fig 6. resnet50 architecture [8].

I built the scripts train.py and train_hook.py, where are developed all the instruction in python 3.x for trained a pre-trained model, and save the model into s3.
So, I did transfer learning by adding the 5 classes needed to solve this problem.
First I have setted the hyperparameter with specific values
hyperparameters = {'batch-size': 64, 'lr': 0.009, 'model-arch': 'resnet50', 'model-n-classes': 5, 'epochs': 6}
and I got the performance of:
Testing Accuracy: 30.225852955309946, Testing Loss: 1.805760314062431
But i tried to improve more my model, so I did Hyperparameter tuning with this values:
hyperparameter_ranges = {
  "lr": ContinuousParameter(0.001, 0.1),

```
      "batch-size": CategoricalParameter([32, 64, 128]),
  }
```

finally, I got this new values:

hyperparameters_tuned = { 'batch-size': 128, 'lr': 0.00254525353426351}

Then I trained the model once more with this new setting.

6. Model evaluation and comparison:

With the Benchmark model I got:

Testing Accuracy: 29.072561268620856

Testing Loss: 1.827670568402486

With my proposal model I got this performance:

Testing Accuracy: 30.033637674195095,

Testing Loss: 1.5222831945588875

As we can observe, The proposal model with resnet50 had a little better performance than the benchmark model (with resnet18).

7. Model Deployment:

Finally, I deployed my model in a sagemaker endpoint, so we can make predictions by invoking it:

```
estimator = PyTorch.attach('smdebugger-capstone-bin-images-final-su-2023-04-16-08-19-08-839')
                              predictor=estimator.deploy(initial_instance_count=1,
                              instance_type='ml.m5.large')
```

I predicted to four images and got these results:

```
inference = predictor.predict(images)
```

array([2, 0, 2, 0]), when labels were tensor([4, 1, 2, 1]).

8. Conclusion

As we could see, the model predictions were not too accurate and the accuracy metric was too low, for this reason we can conclude that the model needs more adjustment. Some hyperparameters could be changed, such as the number of epochs, and we can try by increasing the max_jobs parameter in the HyperparameterTuner object, so it can try with more values (lr and batch-size). Furthermore we can train the model by increasing the computing power, for example by GPU instead of CPUs.

Finally, I realized the dataset was not uniform, because we have differing amounts of images in each class, So , maybe we can use the same number of objects in each class and got an improved model.

References

[1] What is computer vision?. [Online]. Available: https://www.ibm.com/topics/computer-vision

[2] What is image recognition and computer vision?. [Online]. Available: https://www.ibm.com/topics/computer-vision .

[3] Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A Survey of the Recent Architectures of Deep Convolutional Neural Networks. Artif. Intell. Rev. 2020, 53, 5455–5516.

[4] Liu, X. Recent progress in semantic image segmentation. Artifical Intell. Rev. 2019, 52, 1089–1106.

[5] Deep Residual Networks (ResNet, ResNet50) – 2023 Guide. [Online]. Available: https://viso.ai/deep-learning/resnet-residual-neural-network/

[6] resnet18. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/resnet18.html

[7] Resnet18 Model With Sequential Layer For Computing Accuracy On Image Classification Dataset. International Journal of creative research thoughts (IJCRT). vol 10. May. 2022.

[8] The annotated resnet50. [Online]. Available: https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758.