Autonomous Inventory Monitoring at Distribution Centers

Problem definition

For distribution centers, it is crucial to have an accurate record of its inventory. In addition to counting the items manually, recent advancement of image processing has made automating this task possible. Distribution centers often use robots to move objects in bins as a part of their operations, and the robots can capture bin images that can be used by image processing algorithms to automatically count the number of items in each bin. This project tries to build a deep learning model that takes in a bin image and outputs the number of items in the bin.

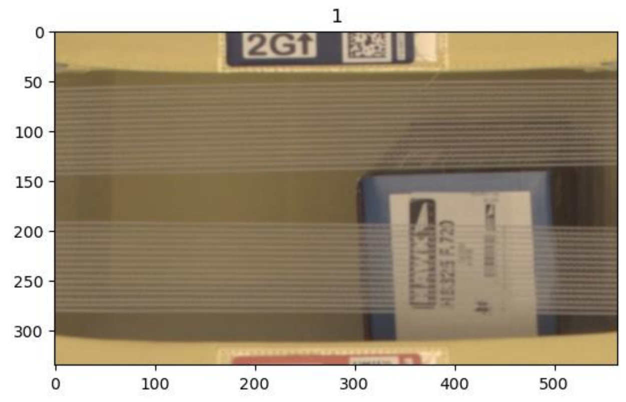
The dataset used for training and evaluating the model is derived from Amazon Bin Image Dataset. It contains over 10,000 images from an operating Amazon Fulfillment Center and each image has 1 – 5 items in it. The bin image, with appropriate preprocessing, is the only input feature. This project applies transfer learning on popular image processing neural networks to classify the images into one of five categories (1 – 5 items). The models are trained on AWS and their performance is evaluated using accuracy, where p_i is the prediction of the ith image and g_i is its label.

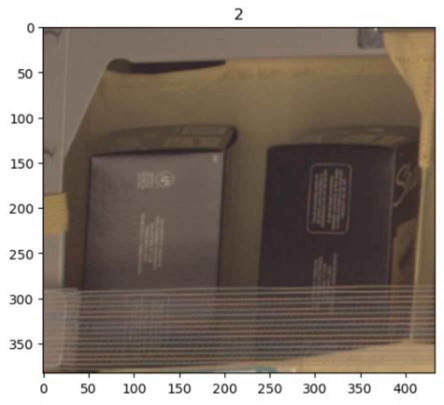
$$\text{Accuracy: } \frac{1}{N}\sum_{i=1}^{N} 1[p_i == g_i]$$

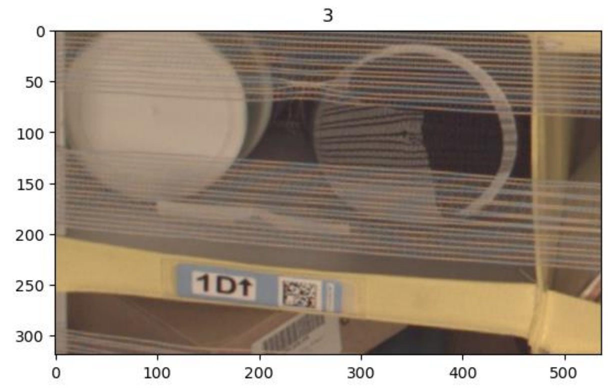
Analysis

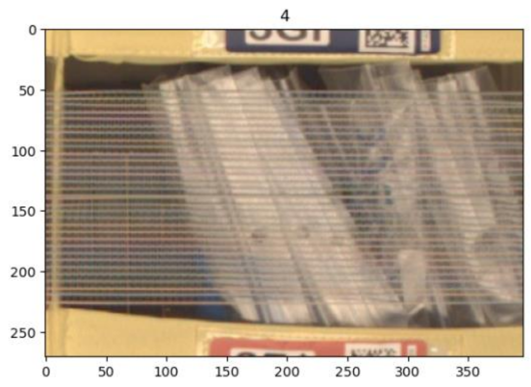
Data Exploration and Visualization

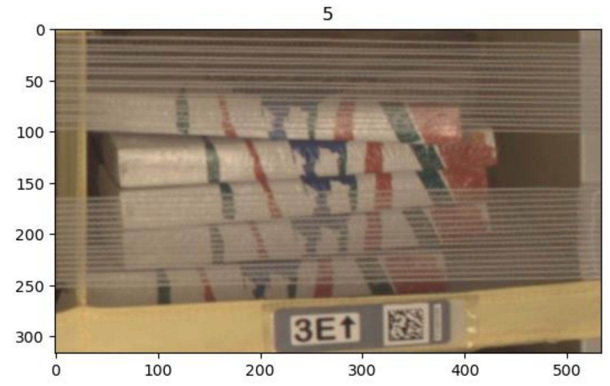
Below are some samples images of each category from the dataset. The tapes on the bins are there to prevent items from falling out and sometimes they can blur the view. Also the images may suffer from low lighting, bad quality or item overlapping, making it hard to count even for human eyes.











The images are all RGB images with three channels, but they have various sizes, therefore resizing is required before passing them to the machine learning model. Their average height is 432 pixels, and average width is 485 pixels. There are less images with 1 and 5 items, but they are not too far away from the other categories and should not interfere with learning. The label distribution is shown in the table below:

Label	1	2	3	4	5
Count	1228	2299	2666	2373	1875

Algorithms and Techniques

Since this is an image classification problem and the dataset is small, it makes sense to take advantage of existing image processing models to gain better performance and shorten training time. This project explores several popular image model structures such as ResNet, reuses their pretrained convolutional layer weights and only trains the last fully connected layer on the bin image dataset. These models are trained on very large datasets for extended periods of time, and their convolutional layers have the ability to recognize general features such as edges and corners. The last fully connected layer is then trained to associate these features with the number of items in each image. This project also does hyperparameter tuning on learning rate and training batch size to achieve the best result.

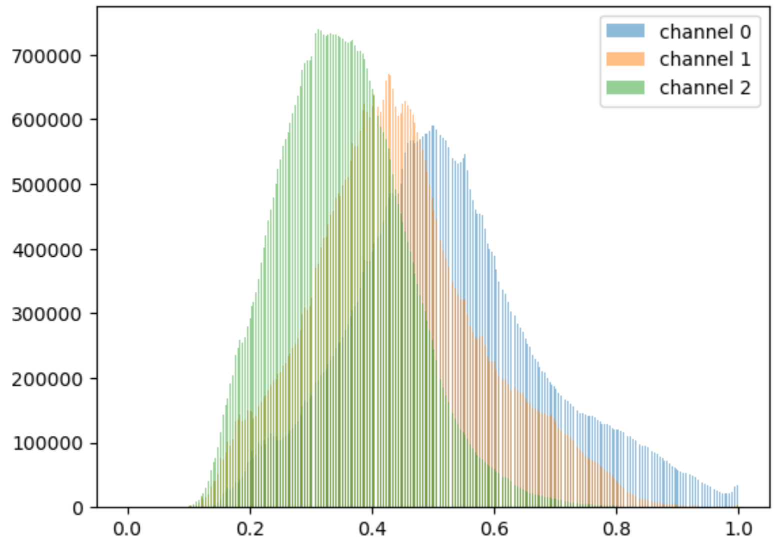
Benchmark

People have been trying to solve the inventory counting problem using image models. For example, Amazon Bin Image Dataset Challenge uses the whole Amazon Bin Image Dataset to train a neural network that can count the number of items in each image. It used a ResNet 34 layer architecture trained from scratch for 40 epochs and achieved accuracy of 55.67%.

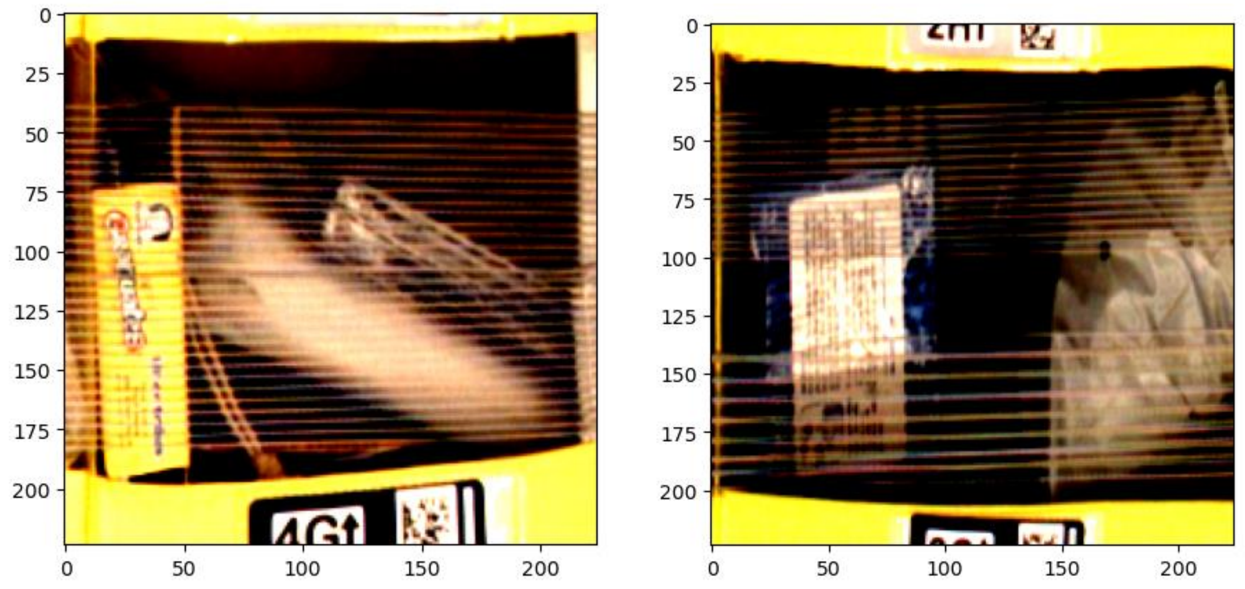
Methodology

Data Preprocessing

The dataset is first divided into train, validation and test set which contains 80%, 20% and 20% of the data respectively. The training set is used for model training. The validation set is used for hyperparameter tuning and the test set is used for final model evaluation and is left out of the training process. Then to help with learning, all images are resized to 224x224 pixels and normalized across each channel before fed into the models. 1000 images are selected randomly to approximate the statistics of the whole dataset. The distributions of tensor values from the 1000 images are shown in the figure below. The means of each channel are (0.52, 0.44, 0.35), and the standard deviations are (0.15, 0.14, 0.105).



Sample images after preprocessing:



Implementation

After preprocessing, the images are passed into models with different architectures and hyperparameter settings to find the best combination. The architectures attempted are ResNet18 and ResNet34. Their pretrained weights are reused except for the last fully connected layer, which is replaced by a new fully connected layer with output size 5 (same as the number of classes) and trained from scratch. The hyperparameters tuned are learning rate (0.0001 – 0.001) and training batch size (64 or 128). All models are trained for 10 epochs, using cross entropy loss and Adam optimizer. All models are trained on AWS sagemaker using p2.xlarge instance. There are six total training jobs attempted and the model with the smallest test set loss is the best model.

Refinement

The best training is then studied closely to find opportunities for improvement. Upon examining the training and validation losses, I found that the model suffers from underfitting so I trained the model for longer to get better results.

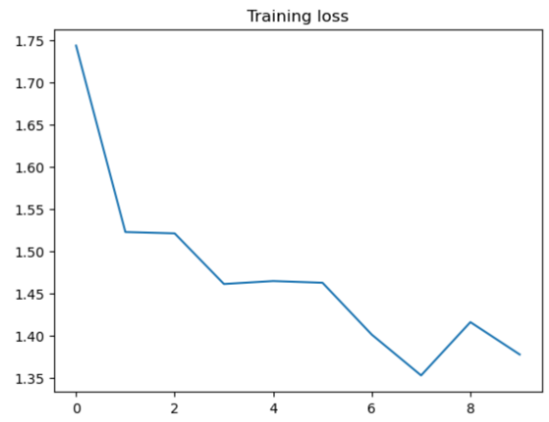
Results

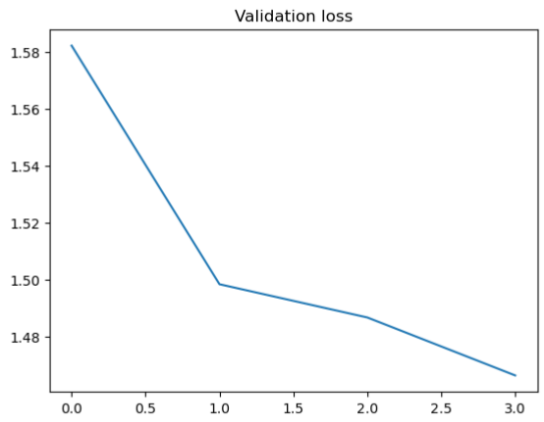
The table below summarizes the performance of each hyperparameter setting. The model with ResNet18 architecture, training batch size 128 and learning rate 0.0001225 has the lowest test loss (1.4628). Its test accuracy is 31.7445%.

batch_size	learning_rate	model_type	Test loss
128	**0.0001225**	**ResNet18**	**1.4628**
64	0.0008797	ResNet34	1.4852
128	0.0005473	ResNet34	1.4914
64	0.0006338	ResNet18	1.4941

64	0.0001634	ResNet34	1.4972
128	0.0007672	ResNet18	1.5279

After studying the training and validation losses of the best training, it looks like both losses are still decreasing at the end of the last epoch and the model may benefit from more training time. Therefore, I restarted the best training with debugger hook enabled and trained the model for 15 epochs this time. Below are the training and validation loss curves of the new training.





Both losses decreased over time, which indicates that there is no overfitting and the model is learning appropriately. The test accuracy now is 32.412%, 2% increase from before.

Conclusions

This project automates inventory monitoring task at distribution centers by using deep neural networks to count the number of items in each bin image. The bin images are first resized and normalized across each channel, then several neural networks are trained on the images to classify them into one of five categories (1 – 5 items). Several pretrained models and hyperparameter settings are attempted to find the best combination. Most weights from the pretrained models are reused, and only their last fully connected layer is retrained on the bin image dataset. The model architectures attempted are ResNet18 and ResNet34. The hyperparameters tuned are learning rate and training batch size. There are six total trainings conducted and the training with the smallest test loss has ResNet18 architecture, learning rate 0.0001225 and batch size 128. It achieves final test accuracy of 32.412%.

There are several things one can explore to further increase accuracy. For example, the whole Amazon Bin Image Dataset can be used for training. More model architectures and hyperparameters can be explored and different level of model weight reuse can be attempted. The models can also be trained for more epochs.