Capstone Project Proposal: Inventory Monitoring Using Object Count Estimation from Bin Images

Tian Gao

June 2025

1. Domain Background

Inventory monitoring is a foundational task in supply chain management, especially in large-scale **automated distribution centers** operated by companies like Amazon, Walmart, and Alibaba. These facilities rely heavily on robotic systems to transport bins that may contain varying quantities and types of items. Errors in bin object counts can lead to shipment inaccuracies, customer dissatisfaction, and logistical bottlenecks.

Traditionally, counting objects in bins has been a manual, error-prone process. With the rise of **computer vision and machine learning**, object counting tasks can now be automated with high reliability. In academic research, object counting is an established problem often addressed using convolutional neural networks (CNNs), especially in domains like crowd counting [1], cell counting [2], and vehicle detection [3]. Applying similar principles to industrial bin images offers a valuable opportunity to reduce labor costs and improve accuracy in logistics workflows.

This project is personally motivating as it represents a **real-world use case** of applying end-to-end machine learning engineering in industry—combining practical skills in AWS SageMaker with academic concepts from deep learning.

2. Problem Statement

The core problem is:

> **Given an image of a bin, accurately predict the number of objects it contains.**

This is a **supervised multi-class classification** problem, where the input is an RGB image and the output is an integer count of objects (e.g., 1 to N). Solving this problem enables automated inventory verification, improving operational efficiency in distribution centers.

The problem is:

- **Well-defined**: Predicting discrete object counts from images.
- **Quantifiable**: Performance can be measured using classification accuracy and confusion matrices.
- **Replicable**: The problem can be solved with standard machine learning frameworks and publicly available datasets.

3. Solution Statement

The proposed solution is to build an **image classification model** that maps bin images to object counts using a **Convolutional Neural Network (CNN)** trained in **AWS SageMaker**.

Steps:

- Use a pre-trained CNN (e.g., ResNet18 or EfficientNet) and fine-tune on bin images.
- Train and validate the model using SageMaker's training jobs and S3-hosted data.
- Evaluate performance using accuracy, precision, recall, and confusion matrix.
- (Optional) Deploy the trained model to a SageMaker endpoint to simulate real-time inference.

The solution is measurable (via classification metrics), scalable (using SageMaker), and reproducible (via training scripts and version-controlled data pipelines).

4. Datasets and Inputs

Dataset: Amazon Bin Image Dataset

- **Size**: 500,000+ RGB images of bins containing 1+ objects.
- **Metadata**: Includes object count, dimensions, object types.
- **Subset**: A small curated subset will be used for training/testing to reduce cost.

Input Features:

- Image pixels (RGB values).
- Optional: metadata such as bin dimensions (may improve model accuracy).

Preprocessing:

- Resize images to a uniform size (e.g., 224x224).
- Normalize pixel values.
- Optional augmentations: rotation, flipping, brightness.

Data Split:

- Training (70%)
- Validation (15%)
- Test (15%)

The dataset is appropriate for the task as it contains sufficient variation in object counts and image quality, simulating real-world scenarios in industrial settings.

5. Benchmark Model

A **baseline model** will be a simple fine-tuned CNN such as **ResNet18** initialized with ImageNet weights. This provides a reliable benchmark for object classification tasks.

The benchmark will help establish a performance baseline (e.g., ∼70–80% accuracy), which more advanced models (e.g., deeper CNNs, ensembles) will aim to surpass.

6. Evaluation Metrics

The performance of both the benchmark and improved models will be evaluated using:

- **Accuracy**: Percentage of correct predictions.
- **Confusion Matrix**: Distribution of true vs. predicted object counts.
- **Precision/Recall/F1**: Especially useful if class distribution is imbalanced.

7. Project Design

Workflow Summary:

1. **Data Setup**:
    - Download subset of dataset using provided starter code.
    - Preprocess and split into train/val/test.
    - Upload to S3 bucket.

2. **Model Training**:
    - Edit `train.py` to implement model architecture and training loop.
    - Run training on SageMaker using Estimator API.
    - Save model artifacts to S3.
3. **Evaluation**:
    - Load model outputs, compute accuracy, confusion matrix.
    - Visualize results using matplotlib/seaborn.
4. **Optional Enhancements**:
    - **Hyperparameter Tuning**: Use SageMaker's HPO jobs to search best learning rate, batch size, optimizer.
    - **Deployment**: Use `predictor.predict()` on a hosted endpoint with sample images.
    - **Cost Optimization**: Use spot instances and monitor resource usage via CloudWatch.
    - **Multi-Instance Training**: Set `train_instance_count > 1` and enable data distribution.

8. Presentation

The final deliverables will include:

- Well-documented code (`train.py`, `sagemaker.ipynb`, etc)
- A completed and well-structured `README.md` with:
    - Problem description
    - How to run the project
    - Evaluation results
- Clear visualizations (confusion matrix, sample predictions)

References

[1] Zhang, Y., Zhou, D., Chen, S., Gao, S., & Ma, Y. (2016). Single-image crowd counting via multi-column CNN. *CVPR*.

[2] Xie, W., Noble, J. A., & Zisserman, A. (2018). Microscopy cell counting with fully convolutional regression networks. *Medical Image Analysis*.

[3] Li, Y., Zhang, X., & Chen, D. (2020). CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. *CVPR*.