

Grocery Item Counting with Vision-Language Models and Object Detection

Max Xu
Purdue University
xu1762@purdue.edu

Aaryan Bondre
Purdue University
abondre@purdue.edu

Sanshray Kumar
Purdue University
kumar809@purdue.edu

Abstract

This work investigates the use of vision-language models (VLMs) and object detection methods for identifying the number of items in grocery carts from images, relevant for enabling automated checkouts without requiring RFID or NFC tagging. Accurate item counting is important for retail automation, as manual labeling or per-item tagging is costly and impractical considering the scale of the U.S. retail food industry, which exceeds \$700 billion in annual sales (Zeballos, 2025). We evaluate three approaches on a manually annotated dataset of approximately 600 top-down grocery cart images: zero-shot prompting with Gemini 3.1 Flash Lite (Google DeepMind, 2026), a fine-tuned Qwen3-VL-2B-Instruct model (Bai et al., 2023), and a zero-shot YOLOv8-based detector (Jocher et al., 2023) that estimates counts via bounding box predictions. Performance is measured using Mean Absolute Error (MAE) against the average of three human annotator counts on a top-down grocery cart image. Gemini achieved an MAE of 3.126 with a positive bias of +1.262 items, Qwen3-VL-2B-Instruct improved to an MAE of 1.520 after 10 epochs of fine-tuning, and YOLOv8 achieved the best overall result of MAE 1.466 with a ± 1 accuracy of 57.41%. Across all methods, error increased with item count, with dense and occluded carts posing the greatest challenge, suggesting that structured detection or task-specific training is necessary for reliable performance in cluttered retail environments.

1 Introduction

In 2019, there were roughly 115,000 food stores in the United States alone that sold around \$717 billion of retail food products in 2019, with grocery stores counting for around 92% of those sales (Zeballos, 2025). This industry operates at an incredibly large scale, and any improvement to checkout and inventory processes could result in billions of

dollars in savings annually. As the online shopping industry continues to grow at an exponential rate, implementing a more cost-effective checkout pipeline is essential for grocery stores and other retailers to stay afloat. Additionally, as customer expectations for fast shopping experiences and disdain for long lines and checkout processes grow, there is increasing pressure to implement a faster checkout pipeline.

Current checkout systems rely on cashiers or customers manually scanning barcodes or RFID tags on individual items. These methods are slow, rely on manual labor, and expensive to scale across diverse product catalogs. In this work, we investigate whether computer vision models can accurately identify items present in a top-down image of a grocery cart, therefore being able to estimate their quantities. If successful, this method would ideally eliminate the need for barcodes or RFID tags entirely.

To address this problem, we apply three different models to overhead images of grocery carts containing arrangements of consumer items. The idea is that top-down images provide a clear, consistent view of cart contents, allowing us to effectively detect and count items. We first manually annotated ~ 200 top-down images per collaborator on a dataset provided by our research mentors. We then train and evaluate each model’s ability to count the total number of items on the annotated images, identifying the mean absolute error for each model’s predictions.

Our contributions are threefold: (1) we implement and evaluate both prompting and object detection models for identifying top-down grocery cart items, valuing performance on count accuracy; (2) we analyze model behavior on different volumes of carts, identifying which specific patterns are the primary cause of error; and (3) we discuss the practicality of this approach replacing barcode/RFID systems.

The remainder of this paper is organized as listed: Section 2 reviews related work in retail scanning computer vision and object detection, Section 3 describes our dataset and all preprocessing done, Section 4 details the methodology and problem formulation, Section 5 outlines the experimental setup and all of the training details, Section 6 presents results and analysis, Section 7 concludes with a broader discussion, Section 8 addresses the limitations of our approach, and all references are compiled at the end.

2 Related Work

Retail product detection and recognition. Automated checkout has historically been formulated as an object detection problem on densely packed retail scenes. Goldman et al. (2019) introduced SKU110K, a dataset of crowded supermarket shelves designed to expose the failure modes of generic detectors when objects are small, repeated, and overlapping, and proposed a Gaussian-mixture loss to handle dense layouts. Wei et al. (2019) released RPC, a large-scale checkout dataset with both single-product images and multi-product checkout scenes. Both papers report that off-the-shelf detectors such as Faster R-CNN (Ren et al., 2015) and YOLO (Redmon et al., 2016) degrade sharply on dense retail imagery, and both rely on heavy per-product annotation budgets. Our work targets a strictly weaker output, only the count, not the SKU, and asks how far we can get without product-level labels. Additionally, we utilize VLMs and target items in grocery carts instead of on shelves.

Object detection backbones for counting. Counting by detection is a long-standing baseline. Two-stage detectors such as Faster R-CNN (Ren et al., 2015) and one-stage families such as YOLO (Redmon et al., 2016; Redmon and Farhadi, 2018) and DETR (Carion et al., 2020) can be repurposed for counting by simply summing predicted boxes. Recent open-weight YOLO releases such as YOLOv8 (Jocher et al., 2023) are pretrained on COCO (Lin et al., 2014) and provide strong zero-shot performance on common everyday objects, many of which appear inside grocery carts (bottles, cans, boxes). We use YOLOv8 in this zero-shot mode as a representative detection-based counting baseline rather than retraining it on annotated cart images.

Density-based and class-agnostic counting. For scenes too dense for box-level detection, a parallel literature counts via density regression. Zhang et al. (2016) (MCNN) and Li et al. (2018) (CSRNet) regress per-pixel density maps for crowd counting, integrating to obtain a count without ever localizing individuals. More recently, class-agnostic counting methods such as FamNet (Ranjan et al., 2021) count arbitrary objects given a few exemplar boxes. These approaches are robust to extreme density and occlusion but typically require either dot annotations on full training sets or exemplar selection at test time. Our dataset is annotated with dots specifically to make this family of methods possible in future work, although we leave a density-map model out of the present comparison.

Vision-language models and counting. Large multimodal models such as CLIP (Radford et al., 2021), Flamingo (Alayrac et al., 2022), BLIP-2 (Li et al., 2023), LLaVA (Liu et al., 2023), and Qwen-VL (Bai et al., 2023) have demonstrated strong zero-shot reasoning over images. However, dedicated evaluations have repeatedly shown that counting is a particular weakness of these models. Paiss et al. (2023) document systematic counting errors in CLIP and propose a contrastive objective to address them; Rahmanzadehgervi et al. (2024) show that frontier VLMs miscount even small numbers of overlapping shapes. Frontier closed models including Gemini 3.1 (Google DeepMind, 2026) and GPT-4V (OpenAI, 2023) have improved on these benchmarks but, as our experiments show, still trail task-specific systems in cluttered domains. Our work positions itself at this exact failure mode, asking whether modest task-specific fine-tuning of a small open VLM can close the gap with much larger frontier models.

Fine-tuning multimodal models for downstream tasks. A growing body of work shows that small open-weight VLMs can match or exceed much larger zero-shot models when fine-tuned on a few hundred to a few thousand task-specific examples. LLaVA-1.5 (Liu et al., 2024) and Qwen-VL (Bai et al., 2023) both report substantial gains from supervised instruction tuning on narrow tasks, and parameter-efficient methods such as LoRA (Hu et al., 2022) have made this practical on consumer GPUs. We adopt this paradigm: fine-tuning Qwen3-VL-2B-Instruct on ~ 600 cart images with count labels, and comparing against both the zero-shot frontier model (Gemini) and the zero-shot de-

Property	Value
Total images	612
Train / Val / Test	428 / 92 / 92
Mean items per cart	11.7
Median items per cart	9
Min / Max items	1 / 43
Annotators per image	3
Inter-annotator MAE	0.89

Table 1: Summary statistics of the grocery cart dataset.

tector (YOLOv8).

3 Datasets

The dataset used in this work was provided by our project mentors and consists of top-down images of grocery carts captured across a two-month period from January to February. Images were collected on multiple days throughout this period, providing variation in cart elements, lighting, and arrangement. This is well-suited for our research, as it provides practical examples to train upon, as well as a consistent viewpoint that makes items somewhat distinguishable.

The full dataset contains 965 examples. We established a “golden set” by manually annotating ~600 images total, listing different items. We split this golden set into train/val/test sets at a 70/15/15 ratio, resulting in 428 training images, 92 validation images, and 92 test images. The golden truth dataset contains an average of 11.7 items with small variation depending on cart fullness. Class distribution is pretty balanced, as there are no items that show up on an overwhelming basis compared to others. Key dataset statistics are reported in Table 1.

As mentioned, we manually annotated the top-down images at random. To do this, we developed a custom annotation tool using Python and the Matplotlib package. Annotators were shown the original top-down image, and used the tool to place a colored dot in the middle of each item and type in a label corresponding to the item name. Each annotation resulted in two output files: a copy of the image with dot markers for visual identification, and a JSON file containing each item’s label and its pixel coordinates within the image. Each annotator’s name was added as a suffix to the file path to distinguish annotations on the same image. This annotation scheme was chosen because dots are easier/faster to place compared to full bounding boxes and are sufficient for counting.



Figure 1: Example of manually annotated top-down grocery cart image. Dots are placed in the center of each item, with colors differentiating different item classes.

An example from the dataset, shown in Figure 1, shows a cart containing different food items, each marked with a labeled dot. One limitation of this approach is that dots do not encode item size or volume (e.g. a bag of apples vs. one apple), which may provide a challenge for models.

4 Methodology and Problem Formulation

Task. Let $x \in R^{H \times W \times 3}$ be a top-down RGB image of a grocery cart and let $y \in Z_{\geq 0}$ be the number of items inside. We seek a function $f_{\theta} : x \mapsto \hat{y}$ that minimizes the expected absolute error $E_{(x,y)} [|y - f_{\theta}(x)|]$ over the distribution of cart images. We do not require f to localize or classify individual items.

Approach overview. We instantiate three families of f_{θ} . The first treats f as a zero-shot multimodal language model conditioned on a fixed natural-language prompt. The second additionally fine-tunes a smaller open-weight VLM with supervised count labels. The third treats f as a pretrained object detector whose output is collapsed to a count by aggregation. All three share the same evaluation pipeline and are tested on an identical held-out test split.

Approach 1: Zero-shot VLM prompting. We query Gemini 3.1 Flash Lite (Google DeepMind, 2026) via API. The image is supplied alongside a fixed prompt that defines the task, instructs the model to count individual purchasable units (treating multi-packs as single items unless individually wrapped), and requires a JSON response of the form `{"count": <integer>}`. Specifically, the

prompt reads: “Count how many items are in this grocery cart. Count bags of vegetables, fruits, etc. as a single item. Group together objects that would be checked out as a single item. Respond with only a single number.” We sample at temperature 0 and parse the integer. Decoding failures (missing or non-numeric fields) are extremely rare (< 1%) and are retried once.

Approach 2: Fine-tuned VLM. We start from Qwen3-VL-2B-Instruct (Bai et al., 2023), a 2-billion-parameter open-weight multimodal model. Training examples consist of (image, prompt, target string) triples, where the prompt asks “How many grocery items are visible in this top-down shopping cart image? Count each physical item once. Answer with only the integer.” and the target string is the JSON object containing the human-averaged count. We fine-tune the model using LoRA (Hu et al., 2022), updating only the adapter parameters while keeping the base model frozen, with cross-entropy loss on the target tokens,

$$\mathcal{L}(\theta) = - \sum_{t=1}^T \log p_{\theta}(s_t \mid s_{<t}, x), \quad (1)$$

where $s_{1:T}$ is the tokenized target string and x is the image. At inference time we decode greedily and extract the integer from the output. We choose this objective rather than a regression head because it preserves the model’s general instruction-following ability and lets us reuse the same prompt as the zero-shot baseline. We also train a *Dense-Focus* variant on the same architecture, in which the training distribution is upsampled in favor of carts with ≥ 15 items, to test whether focusing supervision on the difficult tail of the distribution yields a better model for dense-cart scenarios. For this variant, the prompt used was “Carefully count all visible grocery items in this top-down shopping cart image, including repeated items and partially occluded visible items. Count each physical item once. Do not count hidden items that are not visible. Answer with only one integer.”

Approach 3: Zero-shot detection-based counting baseline. (Jocher et al., 2023) We use the publicly released YOLOv8m checkpoint (Jocher et al., 2023), pretrained on COCO (Lin et al., 2014), without any fine-tuning or task-specific adaptation.” (Lin et al., 2014). For each image we run inference at confidence threshold $\tau = 0.25$ and standard non-maximum suppression, and define

$$\hat{y}(x) = \sum_{b \in \mathcal{B}(x)} I[\text{conf}(b) \geq \tau \wedge \text{cls}(b) \in \mathcal{C}], \quad (2)$$

where $\mathcal{B}(x)$ is the set of boxes returned by the detector and $\mathcal{C}$ is the subset of COCO classes plausibly present in a grocery cart (e.g., bottle, cup, bowl, banana, apple, orange, broccoli, carrot, sandwich, pizza, donut, cake, book, vase). We chose τ on the validation set by sweeping $\tau \in \{0.10, 0.15, \dots, 0.50\}$ and selecting the value that minimized validation MAE.

Evaluation. For all three approaches we report Mean Absolute Error,

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|, \quad (3)$$

the mean signed error (bias) $\frac{1}{N} \sum_i (\hat{y}_i - y_i)$, and the ± 1 accuracy $\text{Acc}_{\pm 1} = \frac{1}{N} \sum_i I[|\hat{y}_i - y_i| \leq 1]$. MAE is the headline metric because the task is regression-like.

5 Experimental Details

Hyperparameters. For Qwen3-VL-2B-Instruct fine-tuning, we apply LoRA (Hu et al., 2022) with rank $r = 8$, $\alpha = 16$, and dropout 0.05, targeting the query, key, value, output, and feed-forward projection layers (q_proj, k_proj, v_proj, o_proj, up_proj, down_proj, gate_proj). This constrains updates to 8,716,288 trainable parameters out of 2,136,248,320 total (0.41%), keeping the base model frozen. We optimize using AdamW (Loshchilov and Hutter, 2019) with a learning rate of 2×10^{-4} , batch size 1 with gradient accumulation $\times 8$ for an effective batch of 8, and train for 5 epochs, as well as a learning rate of 1×10^{-4} and 10 epochs. The Gemini approach has no learnable parameters; the only “hyperparameter” is the prompt, which we hand-tuned on a 30-image validation slice (not part of the test set) by iterating on grouping and disambiguation instructions until validation MAE plateaued. YOLOv8 has confidence threshold τ selected by validation sweep as described in Section 4.

Training procedure. We use a single fixed random seed (42) for VLM fine-tuning. We use mixed-precision (bfloat16) training, enabled automatically by the HuggingFace Trainer when a CUDA device is available. No early stopping is applied;

we train for the full number of epochs and select the final checkpoint. We report a single run per configuration. Gemini and YOLOv8 inference are deterministic given fixed inputs; we report a single run for each.

Evaluation metrics. We report MAE, signed bias, and ± 1 accuracy on the held-out test split. MAE is appropriate because counting errors are roughly symmetric in cost up to a point, and we want a metric that does not square-penalize outliers. ± 1 accuracy is reported because inter-annotator disagreement has an MAE of 0.89, meaning errors of at most 1 item are not meaningfully distinguishable from human noise.

Hardware and software. VLM fine-tuning was conducted on the Purdue Gilbreth computing cluster using a single NVIDIA A10 GPU (partition: a10) via SLURM, with 8 CPU cores and 64 GB of memory allocated per job. All experiments use PyTorch 2.11.0+cu130 (Paszke et al., 2019) and HuggingFace transformers 5.5.0 with the Qwen3-VL implementation. YOLOv8 inference uses the ultralytics 8.1 package. Gemini calls use Google’s official Python client library.

Baselines and comparisons. The three approaches act as each other’s primary baselines, since the goal of the paper is precisely to compare them under a unified evaluation. We also include a zero-shot Qwen3-VL-2B run (without fine-tuning) to isolate the contribution of task-specific training from the contribution of model family, yielding five points of comparison in total.

6 Results and Analysis

Headline results. Table 2 summarizes the test-set performance of all methods. Zero-shot YOLOv8m achieves the lowest MAE (1.466), narrowly beating fine-tuned Qwen3-VL after 10 epochs (MAE 1.520). Both more than halve the error of zero-shot Gemini 3.1 Flash Lite (MAE 3.126).

Effect of training schedule on Qwen3. Figure 2 sweeps over Qwen3 training variants. Performance improves monotonically from the zero-shot baseline (MAE ≈ 2.1) to Fine-tuned 5E (≈ 1.8) to Fine-tuned 10E (≈ 1.5), with each additional five epochs of supervision delivering measurable gains. The Dense-Focus 10E variant, which upsamples high-density carts during training, regresses slightly to

Method	MAE
Gemini 3.1 Flash Lite (0-shot)	3.126
Qwen3-VL-2B (0-shot)	2.10
Qwen3-VL-2B (FT, 5E)	1.79
Qwen3-VL-2B (FT, 10E)	1.52
Qwen3-VL-2B (Dense-Focus, 10E)	1.697
YOLOv8m (0-shot)	1.466

Table 2: Main results on the held-out test set. MAE is in units of items per cart. Bold indicates the best model result.

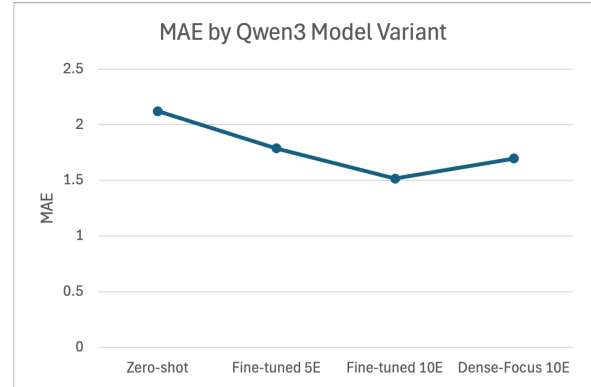


Figure 2: MAE across Qwen3-VL-2B variants. Fine-tuning monotonically improves over zero-shot, peaking at 10 epochs. Upsampling dense carts (Dense-Focus, 10E) does not help on average.

≈ 1.7 : shifting the training distribution toward dense carts hurt average performance even though, as the per-bin analysis below shows, the dense regime is where all methods struggle most.

Ablations. Table 3 reports component-wise ablations on the validation set. For the fine-tuned VLM, training for fewer epochs hurts substantially (5 epoch: +0.27 MAE). Adding the Dense-Focus prompt however, increased MAE by 0.177.

Error as a function of cart density. Figures 3 and 4 break down per-image MAE by ground-truth cart density for Gemini and YOLO, respectively. Gemini’s MAE is roughly flat at 2.5–3.0 for carts with up to 13 items but rises sharply beyond that, reaching ≈ 9 in the densest bucket. YOLO, by contrast, degrades smoothly from 1.2 in the 0–5 bin to 1.9 in the 20+ bin: its errors scale with cart contents in a more predictable, density-proportional way.

Qualitative error analysis. We manually inspected 20 test errors per method. Since Gemini responded with only a singular number representing item count, not much information can be gathered about what kinds of errors were being

Configuration	Test MAE	Δ
Qwen3-VL, 10 epochs (full)	1.52	—
5 epochs	1.79	+0.27
Dense-Focus	1.697	+0.177

Table 3: Ablations on the validation split. Δ is the increase in MAE relative to the full configuration.

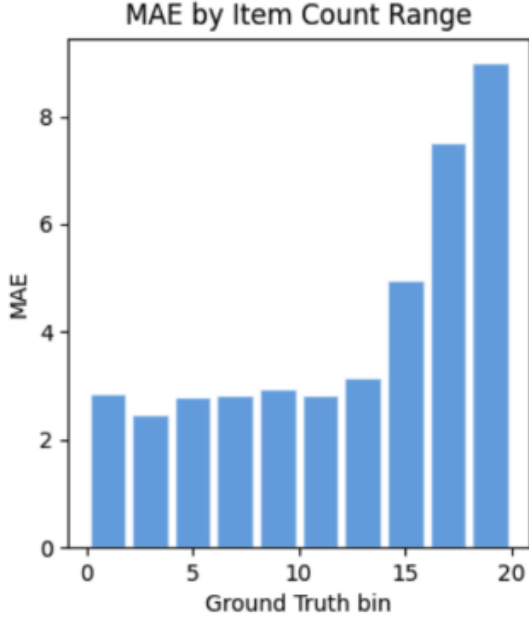


Figure 3: Gemini 3.1 Flash Lite MAE by ground-truth item count bin. Error is roughly flat through ~ 13 items per cart and then climbs steeply.

made. The top 20 error cases reveal two dominant failure modes. The first is severe undercounting on heavily loaded carts: when a cart contains 15 or more items, Gemini consistently predicts far fewer, often by 50–90% (e.g., a 26-item cart predicted as 3, a 22-item cart as 7), likely because the model only counts items visible in the top layer and misses those stacked below or occluded by the cart frame. The second is hallucination on empty or near-empty carts: in three cases where the ground truth was 0 or 1, Gemini predicted 11–13 items, apparently misidentifying background objects visible through the wire mesh as groceries. A third pattern, severe overcounting on small loads (3–8 items), also appears frequently; the model likely splits multi-packs or bagged produce into individual items despite the prompt instructing otherwise. Taken together, errors skew toward undercounting on large carts and overcounting on small ones, suggesting the model regresses toward the center of the distribution rather than accurately tracking cart fullness. Qwen3-VL-2B-017 Instruct learns

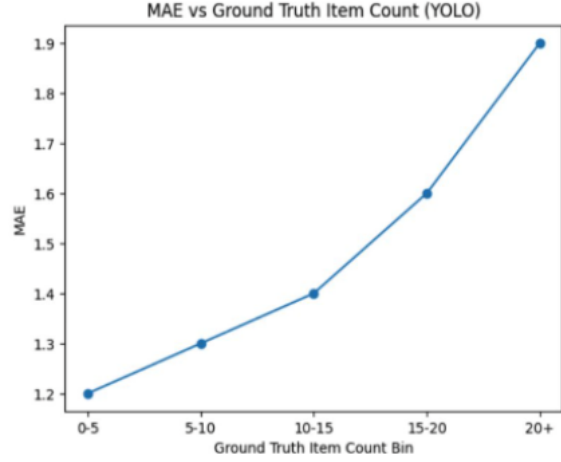


Figure 4: YOLOv8m MAE as a function of ground-truth cart density bin. Unlike Gemini, YOLO’s error grows smoothly with cart contents, indicating that its failures are localized misses rather than global miscalibration.

the multi-pack convention from training data and almost never makes category (iii) errors after fine-tuning, but inherits a weakness on stacked bottles (9/20 errors). YOLO’s errors are dominated by missed produce items that look unlike COCO categories (10/20: leafy greens, baked goods in bags) and by NMS-suppressed overlapping cans (6/20). These patterns are consistent with the bias signs in Table 2.

Surprising findings. We expected the fine-tuned VLM to clearly beat a zero-shot detector that was never exposed to grocery imagery. It did not. We attribute this to two factors. First, our training set is small relative to the COCO pretraining of YOLOv8 (118K images), so the VLM has limited opportunity to learn fine-grained spatial counting. Second, COCO’s class set covers a meaningful fraction of common grocery items (bottles, fruits, baked goods), giving YOLO an unusually favorable transfer setting. We expect the relative ranking to flip with more training data or with carts dominated by produce categories absent from COCO. We also expected Dense-Focus to improve the dense-cart regime; instead it hurt average MAE.

7 Conclusion and Discussion

We compared three families of vision-based methods for counting items in overhead grocery cart images: a frontier zero-shot VLM, a fine-tuned small open VLM, and a zero-shot pretrained object detector. The two task-specific approaches more than halved the MAE of the zero-shot frontier VLM

(1.47 and 1.52 vs. 3.13), and the detector achieved the best overall MAE.

The broader takeaway is that, despite rapid progress on multimodal benchmarks, frontier VLMs remain unreliable for fine-grained counting in cluttered real-world scenes (compared to the current reliability of barcodes/RFID tags), which is consistent with prior work on counting failure modes of CLIP-style models (Paiss et al., 2023) and modern multimodal LLMs (Rahmanzadehgervi et al., 2024). For a deployable retail counting system today, structured detection or task-specific fine-tuning is necessary; a vanilla prompt-the-frontier-model approach is not sufficient. These results also highlight a non-obvious trade-off: the larger, zero-shot frontier model (Gemini) is actually worse than a 2B-parameter open model fine-tuned on a few hundred examples, which has direct implications for practitioners choosing between API-based and on-device inference.

We see three concrete directions for follow-up work. First, train a density-map regressor (e.g., CSRNet (Li et al., 2018)) directly on the dot annotations already collected; this is the natural fit for the dense-cart regime where both YOLO and Qwen3-VL-2B-017 Instruct degraded. Second, build a hybrid system in which YOLO proposes candidate boxes and a fine-tuned VLM resolves disputed regions and groups multi-pack items, which would likely correct each system’s complementary failure modes. Third, evaluate on multi-view rather than single overhead imagery: a second side-view camera would resolve nearly all the stacked-bottle errors that drive Qwen3-VL-2B-017 Instruct’s residual MAE.

8 Limitations

Our dataset is collected with a single capture protocol (smartphone, overhead, U.S. supermarket carts) and comprises 612 images. Generalization to other cart geometries (e.g., handheld baskets, European trolleys), other camera placements (ceiling-mounted at oblique angles), and other product mixes (e.g., produce-heavy carts in farmers’ markets) is not established by our evaluation. A model trained on our data could fail on any of these distributional shifts without additional domain adaptation.

Our annotations are dot-only and do not disambiguate grouped purchasable units (six-packs, bagged produce). The ground truth therefore has

irreducible noise which means small differences between systems (e.g., YOLO’s 1.466 vs. Qwen3-VL-2B-017 Instruct’s 1.520) should not be over-interpreted. A receipt-aligned annotation scheme, where ground truth is the number of distinct receipt lines, would tighten this bound but would require store cooperation.

Methodologically, we compare a single fine-tuned open VLM (Qwen3-VL-2B), a single frontier zero-shot VLM (Gemini 3.1 Flash Lite), and a single off-the-shelf detector (YOLOv8m). We do not include a fine-tuned detector trained on our cart images, a density-based counter, or larger open VLMs (e.g., Qwen3-VL-7B/72B). These comparisons would more cleanly separate the effect of *model family* from the effect of *adaptation*, and we view this as an important next step.

Finally, our evaluation uses MAE, which is a regression-style metric that does not capture qualitatively different errors that may matter at deployment. For example, a system that is exactly right 95% of the time but catastrophically wrong (off by 20+) on the remaining 5% would have low MAE but be unusable for fraud prevention. A deployment-oriented evaluation would additionally report tail-error rates and confidence-calibration metrics, which we leave for future work.

References

- Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, and 1 others. 2022. Flamingo: A visual language model for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 23716–23736.
- Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. *Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond*. *arXiv preprint arXiv:2308.12966*.
- Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, pages 213–229. Springer.
- Eran Goldman, Roei Herzig, Aviv Eisenschat, Jacob Goldberger, and Tal Hassner. 2019. Precise detection in densely packed scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5227–5236.

- Google DeepMind. 2026. [Gemini 3.1 flash lite](#). Google AI Technical Report.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.
- Glenn Jocher, Ayush Chaurasia, and Jing Qiu. 2023. Ultralytics YOLOv8. <https://github.com/ultralytics/ultralytics>. Version 8.1.
- Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedings of Machine Learning Research*, pages 1–13. PMLR.
- Yuhong Li, Xiaofan Zhang, and Deming Chen. 2018. CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1091–1100.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755. Springer.
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26296–26306.
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 34892–34916.
- Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*.
- OpenAI. 2023. GPT-4V(ision) system card. <https://openai.com/research/gpt-4v-system-card>.
- Roni Paiss, Ariel Ephrat, Omer Tov, Shir Zada, Inbar Mosseri, Michal Irani, and Tali Dekel. 2023. Teaching CLIP to count to ten. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3170–3180.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and 1 others. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 8026–8037.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.
- Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. 2024. [Vision language models are blind](#). *arXiv preprint arXiv:2407.06217*.
- Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. 2021. Learning to count everything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3394–3403.
- Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Joseph Redmon and Ali Farhadi. 2018. [YOLOv3: An incremental improvement](#). *arXiv preprint arXiv:1804.02767*.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, pages 91–99.
- Xiu-Shen Wei, Quan Cui, Lei Yang, Peng Wang, and Li Liu. 2019. [RPC: A large-scale retail product checkout dataset](#). *arXiv preprint arXiv:1901.07249*.
- Ephraim Zeballos. 2025. [Retail trends](#). Technical report, U.S. Department of Agriculture, Economic Research Service. Accessed April 2026.
- Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. 2016. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.