WHAT SURVIVES EXTREME COMPRESSION? VISION-LANGUAGE EMBEDDING ALIGNMENT AT MICROCONTROLLER SCALE

Anonymous authors
Paper under double-blind review

ABSTRACT

We ask what remains of CLIP's image–text embedding geometry when ViT-B/32's 87.8 M-parameter image tower is distilled into a 724 K-parameter microcontroller encoder whose class prototypes the teacher's text encoder computes offline. Distilling on CC3M, we find a sharp **capacity–granularity boundary**: the student keeps 39% of teacher accuracy on COCO with classes weighted equally, 12% on ImageNetV2, 5–9% on fine-grained sets and near chance on two robotics sets, a boundary that tracks pretraining coverage. It spans ≈93 effective dimensions where the teacher needs ≈277, and a stronger teacher adds up to 3.2 points at the deployed dimension at no device cost. One coordinate of the *teacher* is its prototypes' shared direction: CLIP ViT-B/32's coordinate 133 carries 79% of the mean prototype's squared norm and enters the prefix only at 256 dimensions, where it adds ≈6 per-image COCO points, all on "person", while class-balanced accuracy falls; projecting it out lifts class-balanced 256d above 128d on COCO and ImageNetV2. Alignment is also teacher-specific: prototypes from any other text encoder score at chance, but a closed-form map fitted on class names alone recovers 64–71% of accuracy offline. The INT8 student runs at 2.5 FPS on an ESP32-S3 and fits an STM32H7 in 1,085 KB flash (464 KB arena, emulated); its 128 px input costs eleven points of top-1 at ten COCO classes (37.9% vs. 48.8%).

1 INTRODUCTION

Vision-language models such as CLIP Radford et al. (2021) place images and text in a single shared embedding space, and every capability they offer downstream - zero-shot classification, retrieval, open-vocabulary detection - is a consequence of that space's geometry. Compression research has asked how much *accuracy* survives when such a model is made smaller Wu et al. (2023); Vasu et al. (2024). We ask a different question: how much of the embedding *geometry* survives, and what remains usable once it has been compressed to the most extreme regime available - a microcontroller.

That regime is a useful extreme precisely because it is unforgiving. As Figure 1 shows, CLIP ViT-B/32's image encoder alone is 87.8 M parameters - 351 MB at FP32, and that is over half the model - against a typical MCU budget of 1–2 MB flash and 512 KB SRAM, and even the smallest published CLIP variants overshoot it by one to two orders of magnitude. Reaching it means compressing that image encoder to 724 K parameters - a factor of 121, or 209× against the full 151 M dual encoder the deployment replaces - which is far past the point where accuracy is preserved, and exactly where the interesting structure appears.

A decoupled deployment makes the question sharp. With the candidate classes known at flash time, prototypes are computed offline with the *full* teacher and only the vision encoder ships, so the student is judged on one thing: whether its visual embeddings still land where the teacher's text embeddings expect them. Accuracy then measures surviving alignment directly; we report it at 224 px (float32) and at 128 px (deployed INT8; Section 4.3).

Existing efficient VLMs do not reach this regime: even with text prototypes precomputed, the smallest released TinyCLIP Wu et al. (2023) and MobileCLIP Vasu et al. (2024) image towers need 8–11 MB at INT8, several times MCU flash, a gap post-training quantization Gholami et al. (2021) alone cannot close. Distilling an image-only student into CLIP's space is itself standard (Popp et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

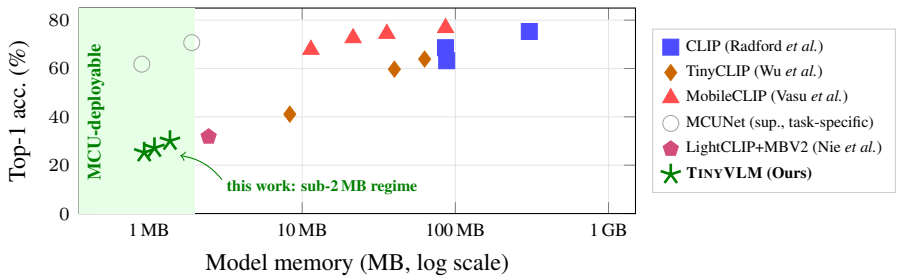

Figure 1: **No prior VLM reaches the sub-2 MB region of the frontier. Axis disclosure:** TinyVLM points are COCO zero-shot (80 classes); the CLIP, TinyCLIP and MobileCLIP points are published ImageNet zero-shot (1000 classes), so this shows frontier *coverage*, not method-vs-method accuracy; on ImageNetV2's 1,000 classes the 0.35× TinyVLM scores 6.3% top-1 at 256d, 12% of its teacher (Section B.15). **Axis:** baselines are *image-encoder* weights at INT8, one byte per parameter; TinyVLM points are total flash (firmware, weights and prototypes; linker-measured for 0.35×, estimated for 0.25× and 0.50×), the strictest convention (seed 42). MCUNet (supervised, not zero-shot) sits at its reported int8 and 4-bit flash Lin et al. (2020). At INT8 the smallest image towers are still 2–11× our flash.

2024), and concept frequency and outlier dimensions are known to shape CLIP at scale (Udandarao et al., 2024; Schrodi et al., 2025); we measure what they do at 121× compression.

We contribute: **(1)** a *capacity–granularity boundary* - the student retains 39% of teacher accuracy on coarse categories with classes weighted equally (COCO, whose per-image accuracy has a 31.6% majority-class floor) but 5–9% on fine-grained ones - relating it to two usually-conflated factors, pretraining coverage, which tracks it, and a rogue teacher coordinate that sharpens it, and showing that, corrected for chance, Core50 behaves like a fine-grained set (Section 4.5). **(2)** an *effective-dimension measurement*: the student spans ≈93 effective dimensions against the teacher's ≈277, and stronger teachers (ViT-B/16; ViT-L/14, ≈375 dimensions) add up to 3.2 points at the deployed 128d, about one width step and free on-device. **(3)** *prototype portability* and its remedy: prototypes from any encoder other than the distillation teacher score at chance, but a closed-form latent-space translation (Maiorca et al., 2023) fitted on class names alone recovers 64–71% of offline, and the residual resists both nonlinear adapters and multi-teacher distillation (Section 4.6). **(4)** an existence proof in hardware - the INT8 student runs at 2.5 FPS on an ESP32-S3 and fits an STM32H7 in 1,085 KB flash and, under emulation, a 464 KB arena, and we measure what its 128 px input costs in accuracy rather than assume it (Section 4.3).

2 Related Work

Vision-language models. CLIP Radford et al. (2021) learns aligned image-text representations from 400M pairs by contrastive training; subsequent work improves it with larger corpora Cherti et al. (2023); Schuhmann et al. (2022), better objectives and training recipes Zhai et al. (2023); Tschannen et al. (2025); Sun et al. (2023), and richer architectures Yu et al. (2022); Li et al. (2023). Open-vocabulary detectors Zhu & Chen (2024); Minderer et al. (2022; 2023); Cheng et al. (2024); Liu et al. (2024) extend the idea to localization; even the smallest of these, YOLO-World-S, has 13 M parameters with its vocabulary folded into the weights (77 M with its text encoder), far beyond MCU flash.

Efficient vision-language models. TinyCLIP Wu et al. (2023) uses cross-modal affinity mimicking and weight inheritance, its smallest ViT-8M/16 variant reaching 41.1% ImageNet zero-shot at 11–23.4 M parameters. MobileCLIP Vasu et al. (2024) designs efficient encoders with multi-modal reinforcement, spanning 53.8 M (S0) to 149.7 M (B) parameters at 67.8–76.8%. CLIP-Lite Shrivastava et al. (2023) makes the contrastive objective cheaper, needing one negative per positive pair, and MobileCLIP2 Faghri et al. (2025) further improves the reinforced-training recipe. *Small* LLM-based VLMs Vasu et al. (2025); Marafioti et al. (2025) still have 0.25B–7B parameters and need ≈0.8 GB of inference memory even at the low end. Closest to our vision-backbone scale, LightCLIP Nie et al.

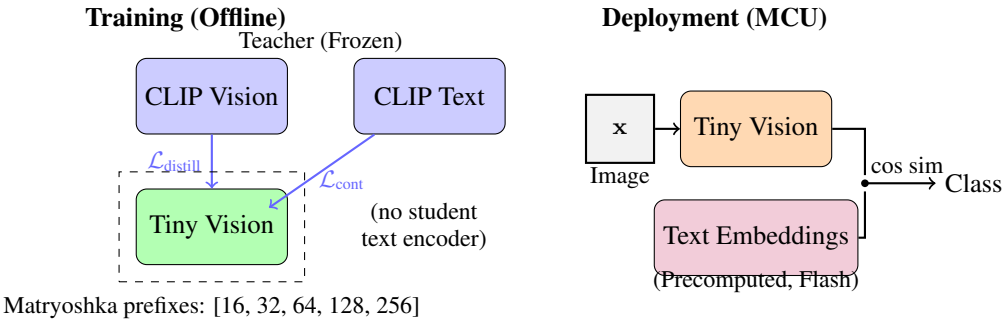

Figure 2: **TinyVLM overview.** (Left) Only the vision path is trained. The frozen teacher supplies both the logit-distillation target ($\mathcal{L}_{\text{distill}}$) and the text side of every contrastive term ($\mathcal{L}_{\text{cont}}$, $\mathcal{L}_{\text{mat}}$), truncated to the student's dimension; there is no student text encoder to train, and the Matryoshka objective is applied to prefixes of the student's 256-d embedding. (Right) On the MCU only the tiny vision encoder runs; class prototypes, computed offline by the teacher, sit in flash.

(2023) pairs a 2.45 M-parameter MobileNetV2 encoder with an 8-layer text Transformer, reaching 31.8% ImageNet zero-shot under YFCC15M-V2 pretraining; its text tower is an order of magnitude larger than that encoder (Table 5). None reaches the MCU regime: even their smallest image towers are 2–11× our 1,085 KB footprint at INT8. Like Popp et al. (2024), TinyVLM trains no student text tower; it differs in serving multiple accuracy-memory operating points from one model via Matryoshka embeddings Kusupati et al. (2022), and in measuring on hardware, not by GPU proxy.

Distillation and TinyML. Knowledge distillation Hinton et al. (2015) transfers from large teachers to compact students. For VLMs, CLIP-KD Yang et al. (2024) finds simple embedding MSE surprisingly effective, CLIP-CID Yang et al. (2025) adds cluster-instance discrimination, and ComKD-CLIP Chen et al. (2024) performs comprehensive feature alignment - all targeting 10–100 MB mobile or server budgets. Distilling to 0.5–1 MB is a different regime, and we characterize what survives it. On the systems side, MCUNet Lin et al. (2020) co-designs architecture and inference engine, reaching 61.8% ImageNet at int8 within 320 KB SRAM and 1 MB flash - less SRAM than our 464 KB arena - and 70.7% with 4-bit weights on a 2 MB-flash MCU, but every MCUNet variant Lin et al. (2020; 2021; 2022), along with MicroNets Banbury et al. (2021), TinyissimoYOLO Moosmann et al. (2023) and DSORT-MCU Boyle et al. (2024), is supervised and task-specific Warden & Situnayake (2019): the label set is fixed at training time and cannot be changed without retraining. Matryoshka Representation Learning Kusupati et al. (2022) and its transformer analogue MatFormer Devvrit et al. (2024) train nested representations whose prefixes remain useful; we extend the idea to vision-language distillation under MCU memory constraints. To our knowledge, TinyVLM is the first CLIP-aligned zero-shot classifier measured on MCU hardware.

3 Method

TinyVLM has three parts (Figure 2): a decoupled architecture (Section 3.1), Matryoshka distillation (Section 3.2), and MCU deployment (Section 3.3).

3.1 Problem Formulation and Decoupled Architecture

Let $f_v : \mathcal{X} \to \mathbb{R}^d$ and $f_t : \mathcal{T} \to \mathbb{R}^d$ be vision and text encoders mapping to a shared d-dimensional space. Given candidate classes $\mathcal{C} = \{c_1, \ldots, c_K\}$, zero-shot classification predicts $\hat{y} = \arg\max_{c \in \mathcal{C}} \cos(f_v(\mathbf{x}), f_t(\text{"a photo of } c\text{"}))$. MCU deployment constrains model size ($|f_v| \le M_{\text{model}}$), text embedding storage ($K \cdot d \cdot b \le M_{\text{embed}}$), and SRAM (typically 256–512 KB).

As in standard CLIP zero-shot inference (Radford et al., 2021; Popp et al., 2024), for closed-set recognition - where candidate classes $\mathcal{C}$ are known at flash time - no text encoder need run at inference. We precompute $\{\mathbf{e}_c\}_{c \in \mathcal{C}}$ offline and store them in flash, so only the vision encoder runs on-device. This reduces inference memory to vision activations alone and enables optimized MCU in-

ference engines David et al. (2021). Our vision encoder is $f_v(\mathbf{x}) = \text{Linear}(\text{GAP}(\text{MobileNetV2}(\mathbf{x})))$ with MobileNetV2 Sandler et al. (2018) width multiplier $\alpha = 0.35$ (936 KB in INT8).

Which text encoder produces the prototypes. Decoupling makes this a free choice, and we exercise it: *the deployed prototypes are computed with the CLIP teacher's text encoder*, not with the student's. $\mathbf{e}_c = \text{trunc}_d(f_t^T(\text{prompt}(c)))$, re-normalised after truncation. The student architecture also declares a text tower (12.9 M), per-dimension Matryoshka heads (913 K) and a spatial projection (328 K), but no term of the objective uses them, so they receive no gradient: exactly 724 K of the checkpoint's 14.8 M parameters are trained - the backbone and the 256-d projection - and they are precisely the parameters that ship.

This cuts both ways. Prototype quality is bounded by what runs on a *workstation*, not by what fits in flash, so the reported accuracies measure how well the student's *visual* representation sits in the teacher's embedding space - exactly what a deployment depends on. The cost is a dependency: CLIP must be run once, offline, per vocabulary. Section 4.6 measures how tightly this binds student to teacher.

3.2 Matryoshka Distillation

Rather than training one model per memory budget, we train a single model with Matryoshka embeddings Kusupati et al. (2022) that can be truncated at deployment. With $d_{\max} = 256$ and $\mathcal{M} = \{16, 32, 64, 128, 256\}$, the prefix $\mathbf{e}^{[:d]} = [\mathbf{e}_1, \ldots, \mathbf{e}_d]$ of $\mathbf{e} \in \mathbb{R}^{d_{\max}}$ must be useful on its own.

We distill from a CLIP teacher with $d_T = 512$. The student never produces text embeddings: in every objective below the text side is the *teacher's* embedding truncated to the student's dimension, $\tilde{\mathbf{e}}_{\text{txt}}^{(d)} = \text{trunc}_d(\mathbf{e}_{\text{txt}}^T)$, so the student is trained *into the teacher's coordinate frame*. This is what makes the deployed prototypes usable with no on-device text encoder - and, as Section 4.6 shows, why they transfer to no other encoder.

The total loss is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{cont}} + \alpha_{\text{mat}}\mathcal{L}_{\text{mat}} + \alpha_{\text{distill}}\mathcal{L}_{\text{distill}} \tag{1}$$

Contrastive loss. Symmetric InfoNCE Oord et al. (2018) at $\tau = 0.07$ between $\mathbf{e}_{\text{img}}^S$ and $\tilde{\mathbf{e}}_{\text{txt}}^{(d_{\max})}$. Per-step batches are small (N=32) to hold the teacher on one GPU, so a FIFO queue of Q=4,096 past image and text embeddings supplies extra negatives to both InfoNCE directions: each term scores 32 positives against 32+4,096 candidates. The queued text side is the frozen teacher's and so is exact; the queued image side is the student's own output from earlier steps and is therefore stale, as we use no momentum encoder. Accumulating gradients over 8 steps gives an optimizer batch of 256 but adds no negatives.

Matryoshka loss. The same objective applied to matched prefixes,

$$\mathcal{L}_{\text{mat}} = \sum_{d \in \mathcal{M} \setminus \{d_{\max}\}} w_d \cdot \mathcal{L}_{\text{cont}}\left(\mathbf{e}_{\text{img}}^{S,[:d]},\ \tilde{\mathbf{e}}_{\text{txt}}^{(d)}\right), \tag{2}$$

with queue negatives truncated to d as well. The sum runs over the four *proper* prefixes $\{16, 32, 64, 128\}$ at $w_d = 1/4$ each; the full 256-d embedding is not a term here because it is already the contrastive objective $\mathcal{L}_{\text{cont}}$. This is what makes each prefix usable.

Logit distillation. We additionally match the student's similarity *structure* to the teacher's, $\mathcal{L}_{\text{distill}} = \tau_d^2\,\text{KL}(\text{softmax}(\mathbf{S}^T/\tau_d)\|\text{softmax}(\mathbf{S}^S/\tau_d))$ with $\tau_d = 4$ Hinton et al. (2015), where $\mathbf{S}^S$ scores student images against the teacher's truncated text and $\mathbf{S}^T$ is the teacher's in-batch image-text similarity.

Dimension selection. At deployment we select $d^* = \max\{d \in \mathcal{M} : K \cdot d \cdot b \le M_{\text{embed}}\}$ for K classes at b bytes per value (1 for INT8); 80 COCO classes in a 10 KB budget gives d^*=128. Algorithm 1 gives the full procedure.

3.3 MCU DEPLOYMENT

Prototype quantization. Prototypes use per-coordinate symmetric quantization:

$$\mathbf{e}_{c,j}^{\text{int8}} = \text{round}\left(\frac{\mathbf{e}_{c,j}}{s_j} \cdot 127\right), \qquad s_j = \max_{c \in \mathcal{C}} |\mathbf{e}_{c,j}| \tag{3}$$

One scale s_j per *coordinate*, shared across classes. This matters: a per-*row* scale ($s_c = \max_j |\mathbf{e}_{c,j}|$) lets uneven norm inflation act as a per-class bias once the dequantised prototypes are compared by cosine similarity, which costs 16.4 points at INT4 (Table 7). Under per-coordinate scales, storage is lossless to INT4 at every dimension - an 8× reduction over float32 - and INT2 costs 0.8 points at 256d. Symmetric (zero-point free) quantization suffices because the post-training distribution is approximately zero-mean ($|\text{mean}|/|\text{std}| < 0.08$ across all 256 dimensions). At inference we dequantize on-the-fly and re-normalise before the argmax.

Inference pipeline. On-device inference (1) resizes the input to 128×128 and quantises it to INT8, with CLIP's input normalisation folded into the first layers of the graph, (2) runs the INT8 MobileNetV2, (3) truncates to the first d^* dimensions, (4) takes cosine similarity with the dequantised, re-normalised prototypes, and (5) returns the argmax. Flash holds the 936 KB INT8 model, the $K \times d^*$ INT8 prototypes and d^* FP32 scales; SRAM holds the activation arena (Section 4.3; layout in Figure 3).

4 EXPERIMENTS

We measure what survives compression: how much teacher accuracy the student retains across six benchmarks and five embedding dimensions, why it fails where it does, whether the alignment transfers to another encoder, and whether it runs on hardware.

4.1 EXPERIMENTAL SETUP

Datasets. We train on Conceptual Captions 3M (CC3M) Sharma et al. (2018) image-text pairs (854K successfully downloaded images) and evaluate zero-shot on COCO Lin et al. (2014) (80 classes), LVIS Gupta et al. (2019) (1203-class long-tail), the fine-grained Flowers102 Nilsback & Zisserman (2008) and Food101 Bossard et al. (2014), Core50 Lomonaco & Maltoni (2017) (10 classes of domestic objects), and TiROD Pasti et al. (2026) (13 classes filmed by a small mobile robot).

Teacher Model and Evaluation Protocol. We use CLIP ViT-B/32 Radford et al. (2021) as the teacher (512-dim embeddings, 151M parameters). Class prototypes for every reported result are computed with the teacher's text encoder (Section 3.1), prompt-ensembled over four generic templates, the same for every dataset (Section A.2), then L2-normalised, truncated to the target dimension and re-normalised. Every reported accuracy is for the float32 student with images resized to 224×224 without cropping, as in training, unless stated otherwise; CLIP's reference short-side resize and centre crop moves the teacher by at most 0.5 on COCO and 2.8 on ImageNetV2, and the student by under 1 (Section B.15). COCO and LVIS are scored per image: the whole image is labelled with its most-annotated category and is correct if that class is predicted top-1. Deployed firmware uses 128 px (Table 15).

Student, training and platforms. The student is MobileNetV2-0.35× with a linear projection to 256 dimensions (Section 3.1), trained for 100 epochs with AdamW (lr 10^{-3}, weight decay 0.01, cosine schedule, 10 warmup epochs) at an optimizer batch of 256 (32-pair steps, 8-step accumulation), $\tau = 0.07$ and $\alpha_{\text{mat}} = 0.5$ (full configuration in Table 3). We deploy on STM32H7 (ARM Cortex-M7 @ 480 MHz, 2 MB flash, 1 MB SRAM) and ESP32-S3 (Xtensa LX7 @ 240 MHz, 8 MB flash, 512 KB SRAM, 8 MB PSRAM).

4.2 MAIN RESULTS

Table 1 reports TINYVLM against its teacher and chance. Despite an orders-of-magnitude smaller footprint, TINYVLM reaches $27.1 \pm 0.6\%$ on COCO and $8.1 \pm 0.1\%$ on LVIS (52% of teacher)

Table 1: Zero-shot classification accuracy (%) across benchmarks. TINYVLM keeps much of the teacher on general objects (COCO, LVIS), is weak on the fine-grained[‡] sets, and is near chance on the robotics sets (Core50, TiROD[◇]). TINYVLM values are mean ± population std across 3 seeds (42, 123, 456); bold marks a best-in-column lead larger than twice the larger spread, so columns where the dimensions are within noise of each other carry no bold; standard deviations are shown in small type. Float32 student at 224 px; the deployed INT8 pipeline at 128 px is in Table 15.

Method	Memory	COCO	LVIS	Core50	TiROD[◇]	Flowers[‡]	Food[‡]
CLIP ViT-B/32 (teacher)[†]	605 MB	35.8	15.6	46.0	29.4	61.3	81.3
Random chance	—	1.3	0.1	10.0	7.7	1.0	1.0
Majority class[§]	—	31.6	1.4	—	—	—	—
TINYVLM (256d)	936KB	**27.1**$_{\pm 0.6}$	**8.1**$_{\pm 0.1}$	11.5$_{\pm 0.5}$	7.8$_{\pm 2.2}$	5.2$_{\pm 0.6}$	4.0$_{\pm 0.4}$
TINYVLM (128d)	936KB	20.4$_{\pm 0.5}$	7.4$_{\pm 0.1}$	11.1$_{\pm 0.8}$	4.7$_{\pm 1.8}$	5.3$_{\pm 0.2}$	4.8$_{\pm 0.3}$
TINYVLM (64d)	936KB	16.5$_{\pm 0.3}$	5.9$_{\pm 0.1}$	10.4$_{\pm 0.1}$	5.4$_{\pm 1.6}$	4.8$_{\pm 0.2}$	5.3$_{\pm 0.1}$
TINYVLM (32d)	936KB	13.3$_{\pm 0.2}$	3.8$_{\pm 0.2}$	10.1$_{\pm 0.7}$	5.0$_{\pm 1.5}$	4.6$_{\pm 0.3}$	4.2$_{\pm 0.3}$
TINYVLM (16d)	936KB	8.9$_{\pm 0.1}$	2.2$_{\pm 0.1}$	9.8$_{\pm 0.6}$	6.3$_{\pm 0.8}$	2.9$_{\pm 0.2}$	2.6$_{\pm 0.2}$

[†]Teacher and student are measured under the identical 4-template protocol on the same validation splits (Section 4.1). [§]Always predicting the most frequent label (COCO: "person"; LVIS: "zebra"); computed for COCO and LVIS. [‡]Fine-grained, minimal CC3M coverage (Section 4.5). [◇]Imagery from a small mobile robot's camera; the student is at chance, with a spread above its distance from chance. Memory: INT8 TFLite model; prototypes stored separately (Table 14).

at 256 dimensions. Per-image COCO accuracy has a high floor: 31.6% of val images are labelled "person", so always guessing it beats the student. The student predicts "person" for 16.8% of images at 57% precision (teacher: 1.8%, at 44%), so its 9.6 "person" points are recognition, not base-rate credit, but on a class the teacher almost never predicts. Weighting all 80 classes equally removes this: 21.2% against 54.9%, a retention of 39%, our headline (seed 42; 40% at the deployed 128d, which coordinate 133 does not reach; Section B.15). The robotics benchmarks are near chance (Core50 11.5 vs. 10.0, TiROD 7.8 vs. 7.7) and the fine-grained sets (Flowers102, Food101) are weak; Section 4.5 separates the factors. TinyCLIP and MobileCLIP report only ImageNet zero-shot and COCO *retrieval*, so they appear only in Figure 1; on ImageNetV2's 1,000 classes TINYVLM reaches 6.3% (7.0% at 128d), 12% of its teacher's 52.6% and far below their numbers (Section B.15).

Pseudo-label baseline. A CLIP-pseudo-label supervised baseline with identical backbone and budget reaches 24.5% on COCO against 27.8% for our student at the same seed (42). The margin is modest (+3.3 pp); the structural difference is not (Section 4.7).

Matryoshka vs. naive truncation. Training with the Matryoshka objective makes the prefixes of the student's embedding usable in their own right: it beats naive truncation of CLIP's own embedding at every dimension up to 128d (2.1× at 16d), though not at 256d, where truncated CLIP keeps half its 512-d space (30.3 vs. 27.1; Figure 4, Section B.7).

4.3 MCU BENCHMARKS

On ESP32-S3 at 240 MHz the MBV2-0.35× student runs 396.7 ms end-to-end (2.5 FPS) with a 464 KB TensorFlow Lite Micro arena in external PSRAM and a 1,288 KB application image. Measured with a Nordic PPK2, inference draws 125.4 mW above a 130.3 mW idle, or 49.8 mJ per image; the usual datasheet figure (181.5 mW) overstates that increment by 45%. The STM32H7 board failed before this graph could be timed; building the same firmware for it gives 1,074 KB of flash (1,085 KB with the 256-dim table), and running that binary under emulation gives a 464 KB arena, which fits the 512 KB AXI SRAM; the emulator reproduces the board's earlier hardware-measured arena to the byte. Scoring 80 prototypes at 128d takes 790 μs (0.2% of inference), and up to K=5,000 fits the remaining flash (Table 17). At 224 px peak SRAM would reach ≈1.4 MB, beyond either board, and resolution is where accuracy is lost: at 128 px the pipeline reaches 16.0% COCO top-1 at 256d

and 37.9% at $K=10$, against 28.1% and 48.8% for the float32 student at 224 px on the same images (grid-control seed-42 checkpoint of Table 18; 2,047 COCO val images), and INT8 accounts for under a point of that gap (Table 15). Full per-platform figures, including a MAX78000 build of a smaller network that carries none of these accuracy results, are in Section B.9.

4.4 Ablation Studies

Ablations use seed 42 (100 CC3M epochs each; tables in Section B). In brief: logit distillation and the Matryoshka objective each lift 64d from 7.1 to 15.7 or 16.5 when added alone, but distillation adds nothing once the Matryoshka term is present (its target is the teacher's full 512-d similarity, which a student scored in the truncated space cannot match); the width effect differs between harnesses ($+1.2$ at 256d from 0.25× to 0.50× here, $+4.8$ in the grid of Table 18) and costs ≈0.44 MB of flash; inverse dimension weighting is within seed noise; data scales sub-linearly (500K = 94% of 854K); and prototypes are lossless at INT4 (Table 7).

Stronger teachers. Over the 3-teacher × 3-width grid (seed 42; Table 18), ViT-B/16 and ViT-L/14 ($+5.4$ and $+12.1$ ImageNet points over ViT-B/32) add at most 3.2 and 1.3 points at the deployed 128d, and trail at 256d mainly because ViT-B/32 alone gains its coordinate 133 there (paragraph (ii) of Section 4.5). At 128d ViT-B/16's gain is about one width step ($+3.2$/$+2.4$ against $+2.5$/$+2.0$ at 0.25/0.35×, one seed) at no device cost; we have not redeployed with it. Gains of about one width step fit a capacity limit: on 2K COCO val images ViT-L/14 needs ≈375 PCA-95 dimensions (components for 95% of variance) and ViT-B/32 ≈277, while the student spans only ≈93 (Figure 5). Capacity is not separated from data (one width, one data scale), and PCA-95 may understate a Matryoshka-trained student (Figure 5).

4.5 Where the Student Fails

Teacher retention at 256d falls with category granularity: COCO 39% class-balanced; per image, LVIS 52%, Flowers 9%, Food 5%. We separate two factors often conflated under "fine-grained is hard", then correct the robotics sets for chance.

(i) Pretraining coverage. Table 19 counts how often each dataset's class names appear in the 854K CC3M captions. COCO classes appear in 0.23% of captions on average against 0.002% for Flowers102 - a 100× gap that tracks the COCO–Flowers retention gap, as concept frequency does for CLIP at scale (Udandarao et al., 2024; Parashar et al., 2024); within COCO it also tracks per-class accuracy (Spearman $+0.38$, Section B.16). Low coverage and fine granularity coincide here and cannot be separated; Core50's high coverage is no counterexample, since its names ("can", "ball") are polysemous and word matches overcount them, and its "glass" prototype names drinkware where the class is eyewear (Section B.15).

(ii) One teacher coordinate: the prototypes' shared direction. On COCO per-image accuracy rises monotonically with dimension (8.9→27.1 from 16d to 256d); on the fine-grained sets it does not - Food101 peaks at 64d and Flowers102 flattens after 128d. CLIP ViT-B/32 coordinate 133 carries ≈24× the median coordinate amplitude, and since $128 < 133 \leq 256$ it enters the Matryoshka prefix at exactly one dimension. Zeroing it (dataset-specific prompts, Section B.12) reverses Food101's $128 \rightarrow 256$ drop ($-1.59 \rightarrow +0.35$) and erases ≈70% of COCO's per-image step ($+7.44 \rightarrow +2.24$). The coordinate is the prototypes' common direction - 79% of the mean 256-d COCO prototype - and it pulls predictions toward generic classes. In the deployed checkpoint COCO's step is one class, "person" (3.4 to 9.6 points, more than the whole $+6.0$), while class-balanced accuracy falls (21.8 to 21.2); zeroing the coordinate moves 18% of predictions, 8.3 points of them off "person", and projecting the mean direction out of images and prototypes lifts class-balanced 256d to 22.4, above 128d. On ImageNetV2, which the account was not fitted to, zeroing lifts 256d from 6.3 to 7.4, above 128d's 7.0 (Section B.15). This is anisotropy of the *teacher's* space - rogue dimensions (Timkey & van Schijndel, 2021; Sun et al., 2024), the modality gap (Liang et al., 2022; Schrodi et al., 2025), the common mean that all-but-the-top removes (Mu & Viswanath, 2018) - not the student's spare capacity. On the deployed INT8 pipeline, masking coordinate 133 in the prototype table, a free firmware change, raises class-balanced accuracy from 12.9 to 13.5 but lowers per-image from 16.0 to 12.2; we ship the unmasked table and report both (Table 15). Changing the

teacher agrees: ViT-B/16 and ViT-L/14 place their dominant coordinates outside any 256-d prefix, and over a 3-teacher $\times$ 3-width grid COCO's per-image $128 \rightarrow 256$ step is $+6.0$ to $+7.0$ under ViT-B/32 but at most $+0.8$ otherwise (Section B.12; not re-scored class-balanced). The rule: *check whether a truncation prefix admits the prototypes' mean direction*, not tune dimension per dataset.

(iii) The robotics sets, corrected for chance. Raw retention mixes chance levels from 0.1% to 10%. Measured as $(s-c)/(t-c)$ for student s, teacher t and chance c, retention is 75% on COCO (37% class-balanced, the corrected form of the 39% headline; per image its floor is the 31.6% majority class, not 1.3%), 52% on LVIS (majority class 1.4%), 7% on Flowers102, 4% on Core50, 4% on Food101 and 0.5% on TiROD. Core50 is therefore not anomalous: it sits with the fine-grained sets, and its close-up, hand-held objects are visually fine-grained even where their names are common; five of its ten prototypes take 98% of the predictions (Section B.16). TiROD's student is at chance with a spread larger than its distance from chance (7.8 ± 2.2 vs. 7.7), so that column carries no signal; its teacher reaches 29.4% (protocol: Section B.15).

Operating points. Over 20 random vocabularies per size at 256d (3-seed mean), top-1 rises from 27.1% at $K{=}80$ to 46.6%, 48.9% and 65.3% at $K{=}20$, 10 and 5, and top-3 reaches 76.4% at $K{=}10$ (Table 16); the deployed INT8 pipeline at 128 px reaches 37.9% at $K{=}10$ (seed 42). TinyVLM is a coarse-category instrument for small, COCO-like vocabularies.

4.6 Prototype Portability, and How Much of It Can Be Undone

Because prototypes are minted offline (Section 3.1), must they come from the distillation teacher? If the student learned a *general* CLIP-like space, another encoder's prototypes should transfer; if it learned one teacher's basis, they should not. Holding the student fixed and swapping only the encoder, nothing transfers. Prototypes from CLIP ViT-L/14 and from two OpenCLIP models trained on LAION-2B all score at or near the 1.3% random baseline, against 27.8% when the student's own teacher mints the prototypes (Table 2, 'raw'; seed 42), at every Matryoshka dimension (Section B.8). The most informative case is OpenCLIP ViT-B-32, which shares the teacher's architecture *and* embedding width and differs in training data and recipe - it too collapses to chance. Architecture is not the binding constraint; the specific learned basis is. Independently trained encoders are known to differ by roughly a linear change of basis (Moschella et al., 2023; Maiorca et al., 2023), which no image tower absorbs without a map, so alignment is a property of a teacher–student *pair*, not of a model.

That lock-in is a deployment problem - a fielded device cannot adopt a newer or multilingual encoder without redistilling - and most of it can be undone *without touching the device*. Since the student is trained into the teacher's truncated text space (Section 3.2), that space *is* its coordinate frame, and adopting an encoder f_t^2 reduces to a regression into that frame,

$$\mathbf{W}^\star = \arg\min_{\mathbf{W}} \sum_{c \in \mathcal{V}} \left\| \mathbf{W}\, f_t^2(c) - \mathrm{trunc}_d\big(f_t^T(c)\big) \right\|_2^2 + \lambda \|\mathbf{W}\|_2^2, \tag{4}$$

a ridge problem with a closed form. It needs only *class names* - no images, no labels, no training - and $\mathbf{W}$ is applied on the workstation when prototypes are minted, so the on-device cost is exactly zero. Fitted on 983 ImageNet-1K class names with the evaluation vocabulary removed by exact name match - a lexical split, so ImageNet hyponyms of COCO classes remain in the fitting set - a single linear map lifts every foreign encoder from chance to 64–71% of oracle, the accuracy with the teacher's own prototypes (Table 2). Capacity does not appear to be the bottleneck: a two-layer MLP, early-stopped on held-out vocabulary and averaged over three fits, is mixed rather than better (-1.7, $+1.4$, -0.2 against the linear map), so a nonlinear parameterisation buys nothing consistent. Whether the residual third reflects the student's $\approx$93 dimensions or the map itself is untested.

4.7 Extending a Deployed Vocabulary

The margin over a supervised head is small ($+3.3$ pp at seed 42), so we measure what text prototypes buy: a device ships with 70 COCO classes and ten new ones must be added in the field. For TinyVLM the vision encoder is vocabulary-independent: minting ten prototypes takes **0.13 s** and adds **2.5 KB** of flash at 256d, with no training and no reflash of the model, and reaches **35.3%** top-1 on the 851 held-out images of those ten categories (the last ten COCO classes, toaster to toothbrush),

Table 2: Adapting a foreign text encoder, COCO top-1 (%) at 256d. **raw**: the encoder's prototypes truncated, as in Section 4.6. **adapted**: after a closed-form map $\mathbf{W}$ fitted on 983 class names that share no name with evaluation. **multi-teacher**: a control student distilled against two text spaces at once, scored against its own oracle. **Seed 42** (first row, 3 seeds: 27.1 ± 0.6); chance 1.3.

Prototype source	raw	adapted	multi-teacher
CLIP ViT-B/32 *(own teacher)*		**27.8**	15.0
CLIP ViT-L/14	0.7	**19.5**	11.1
OpenCLIP ViT-B-16 (LAION-2B)	1.2	**17.8**	13.1
OpenCLIP ViT-B-32 (LAION-2B)	1.4	**19.8**	14.1[†]

[†]Seen during multi-teacher training, so not a held-out test for that column.

scored among the ten new prototypes alone, against a 10% baseline. For the supervised head the problem is structural: a 70-way softmax places zero probability mass on the new classes, so *every* novel-class image is misclassified. The asymmetry is data: extending the prototype table needs only the ten class *names*, whereas extending any trained head needs images of the new classes, labelled or pseudo-labelled. At fixed backbone and compute the two are close on a *shared* vocabulary and not comparable on one that changes later.

5 Discussion and Conclusion

Compressing CLIP's image encoder 121× preserves a usable but sharply bounded fragment of its embedding space: 39% of teacher accuracy on COCO with classes weighted equally, falling to 12% on ImageNetV2 and 5% on Food101. The student spans only ≈93 effective dimensions of a geometry needing ≈277, consistent with two further results: stronger teachers add at most 3.2 points, and prototypes from other text encoders need an offline map (64–71% of oracle). 48.9% top-1 at ten classes for the float32 student at 224 px, and 37.9% for the deployed INT8 pipeline at 128 px, running at 2.5 FPS on an ESP32-S3 and fitting an STM32H7 in 1,085 KB flash.

Limitations. Fine-grained accuracy is low where pretraining coverage is weak; the system is closed-set and image-level, with no localization; the student is near chance on both robotics benchmarks (Core50 behaves like a fine-grained set, TiROD carries no signal); the measured ESP32-S3 could hold a wider student, since the 121× target is set by the 2 MB-flash STM32H7 class, which we only emulate; the deployed 128 px input costs about eleven points of top-1 against the 224 px evaluation, and neither training at that resolution (Touvron et al., 2019) nor patch-based inference (Lin et al., 2021) was tried; per-seed variance on out-of-distribution data exceeds the inter-dimension gap; foreign encoders recover only 64–71% of oracle; and the dimension-scaling result is weaker than it looks, since COCO's $128 \to 256$ gain is per image, all on one class, while class-balanced accuracy falls. **Takeaway.** Target coarse categories at small K and check them on the target domain, audit class-name frequency in the corpus, try ViT-B/16 as teacher at 128d, and check whether a truncation prefix admits the prototypes' mean direction.

Reproducibility Statement

Every number in this paper is produced by a script in the accompanying code. The training objective actually optimised is stated in Section 3.2 and the deployed-prototype pathway in Section 3.1; the two are easy to conflate, so Section B.15 gives the protocol that separates them. All main-table results use one uniform 4-template prompt protocol on identical splits for teacher and student (Section 4.1), with 3-seed means (42, 123, 456); ablations and the adapter study use seed 42 alone and say so in place. Accuracies are for the float32 student at 224×224 unless stated otherwise; the deployed INT8 pipeline is evaluated separately at its 128×128 input (Table 15). ESP32-S3 and MAX78000 figures are UART measurements on hardware, and the power of both is measured with a Nordic PPK2 (whole board on MAX78000, whose core rail is not exposed; see Section B.9). The STM32H7 board failed before the final graph could be timed: its flash is from the linker and its arena from running the same binary under emulation, which reproduces the board's earlier measured arena to the byte, and no STM32H7 latency is reported. The firmware builds against pinned dependencies

(TensorFlow Lite Micro `bfeb43a`, `esp-tflite-micro` `dd40859`). Section B.11 gives the storage arithmetic in closed form. The prototype adapter is a closed-form ridge fit on class names only, sharing no name with evaluation, and needs no training run to reproduce.

Ethics Statement

This work uses only public datasets (CC3M, COCO, LVIS, Flowers102, Food101, Core50, TiROD, ImageNetV2) and a public pretrained teacher. It involves no human subjects and releases no new data. The relevant risk is a deployment one, and it is the subject of the paper rather than an aside: a model of this size retains coarse object categories and loses fine-grained ones, and a ten-class device can sit near chance on its own target domain, as the student does on Core50. An always-on camera that recognises "person" also invites surveillance uses; the student identifies no one, but deployments should follow local privacy law. Reporting only the COCO number would invite exactly the misuse this paper argues against, which is why the fine-grained and robotics failures are reported beside it and the operating points as means over random vocabularies.

AI Use Statement

Generative AI was used solely for writing assistance, including drafting, editing, and revising the manuscript. It was not used for research, experimentation, data analysis, or verification of results. The authors independently conducted and verified the research and take full responsibility for the final content, including all claims and reported results.

References

Colby Banbury, Chuteng Zhou, Igor Fedorov, Ramon Matas Navarro, Urmish Thakker, Dibakar Gope, Vijay Janapa Reddi, Matthew Mattina, and Paul Whatmough. MicroNets: Neural network architectures for deploying TinyML applications on commodity microcontrollers. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 3, pp. 517–532, 2021.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, pp. 446–461, 2014.

Liam Boyle, Julian Moosmann, Nicolas Baumann, Seonyeong Heo, and Michele Magno. DSORT-MCU: Detecting small objects in real-time on microcontroller units. *IEEE Sensors Journal*, 24 (24):40231–40239, 2024. doi: 10.1109/JSEN.2024.3425904.

Yifan Chen, Xiaozhen Qiao, Zhe Sun, and Xuelong Li. ComKD-CLIP: Comprehensive knowledge distillation for contrastive language-image pretraining model. *arXiv preprint arXiv:2408.04145*, 2024.

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. YOLO-World: Real-time open-vocabulary object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16901–16911, 2024.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2829, 2023.

Robert David, Jared Duke, Advait Jain, Vijay Janapa Reddi, Nat Jeffries, Jian Li, Nick Kreeger, Ian Nappier, Meghna Natraj, Shlomi Regev, Rocky Rhodes, Tiezhen Wang, and Pete Warden. TensorFlow Lite Micro: Embedded machine learning for TinyML systems. *Proceedings of Machine Learning and Systems*, 3:800–811, 2021.

Devvrit, Sneha Kudugunta, Aditya Kusupati, Tim Dettmers, Kaifeng Chen, Inderjit Dhillon, Yulia Tsvetkov, Hannaneh Hajishirzi, Sham Kakade, Ali Farhadi, and Prateek Jain. MatFormer: Nested transformer for elastic inference. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7(1):12140, 2017.

Fartash Faghri, Pavan Kumar Anasosalu Vasu, Cem Koc, Vaishaal Shankar, Alexander Toshev, Oncel Tuzel, and Hadi Pouransari. MobileCLIP2: Improving multi-modal reinforced training. *Transactions on Machine Learning Research (TMLR)*, 2025. arXiv:2508.20691.

Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*, 2021.

Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5356–5364, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 30233–30249, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning (ICML)*, pp. 19730–19742, 2023.

Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Zou. Mind the gap: Understanding the modality gap in multi-modal contrastive representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. MCUNet: Tiny deep learning on IoT devices. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 11711–11722, 2020.

Ji Lin, Wei-Ming Chen, Han Cai, Chuang Gan, and Song Han. MCUNetV2: Memory-efficient patch-based inference for tiny deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, Chuang Gan, and Song Han. On-device training under 256KB memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision (ECCV)*, pp. 38–55, 2024.

Vincenzo Lomonaco and Davide Maltoni. CORe50: a new dataset and benchmark for continuous object recognition. *Proceedings of the 1st Annual Conference on Robot Learning (CoRL)*, pp. 17–26, 2017.

Valentino Maiorca, Luca Moschella, Antonio Norelli, Marco Fumero, Francesco Locatello, and Emanuele Rodolà. Latent space translation via semantic alignment. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Andrés Marafioti, Orr Zohar, Miquel Farré, Merve Noyan, Elie Bakouch, Pedro Cuenca, Cyril Zakka, Loubna Ben Allal, Anton Lozhkov, Nouamane Tazi, Vaibhav Srivastav, Joshua Lochner, Hugo Larcher, Mathieu Morlon, Lewis Tunstall, Leandro von Werra, and Thomas Wolf. SmolVLM: Redefining small and efficient multimodal models. *arXiv preprint arXiv:2504.05299*, 2025.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection. In *European Conference on Computer Vision (ECCV)*, pp. 728–755, 2022.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Julian Moosmann, Marco Giordano, Christian Vogt, and Michele Magno. TinyissimoYOLO: A quantized, low-memory footprint, TinyML object detection network for low power microcontrollers. In *IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pp. 1–5, 2023.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. In *International Conference on Learning Representations (ICLR)*, 2023.

Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations (ICLR)*, 2018.

Ying Nie, Wei He, Kai Han, Yehui Tang, Tianyu Guo, Fanyi Du, and Yunhe Wang. LightCLIP: Learning multi-level interaction for lightweight vision-language models. *arXiv preprint arXiv:2312.00674*, 2023.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, pp. 722–729, 2008.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Shubham Parashar, Zhiqiu Lin, Tian Liu, Xiangjue Dong, Yanan Li, Deva Ramanan, James Caverlee, and Shu Kong. The neglected tails in vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Francesco Pasti, Riccardo De Monte, Davide Dalle Pezze, Gian Antonio Susto, and Nicola Bellotto. TiROD: Tiny robotics dataset and benchmark for continual object detection. *Journal of Intelligent & Robotic Systems*, 2026. doi: 10.1007/s10846-026-02430-9. arXiv:2409.16215.

Niclas Popp, Jan Hendrik Metzen, and Matthias Hein. Feature distillation improves zero-shot transfer from synthetic images. *Transactions on Machine Learning Research (TMLR)*, 2024.

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. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pp. 8748–8763, 2021.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *International Conference on Machine Learning (ICML)*, pp. 5389–5400, 2019.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.

Simon Schrodi, David T. Hoffmann, Max Argus, Volker Fischer, and Thomas Brox. Two effects, one trigger: On the modality gap, object bias, and information imbalance in contrastive vision-language models. In *International Conference on Learning Representations (ICLR)*, 2025.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:25278–25294, 2022.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Association for Computational Linguistics (ACL)*, pp. 2556–2565, 2018.

Aman Shrivastava, Ramprasaath R. Selvaraju, Nikhil Naik, and Vicente Ordonez. CLIP-Lite: Information efficient visual representation learning with language supervision. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 8433–8447, 2023.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *Conference on Language Modeling (COLM)*, 2024.

Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. EVA-CLIP: Improved training techniques for CLIP at scale. *arXiv preprint arXiv:2303.15389*, 2023.

William Timkey and Marten van Schijndel. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4527–4546, 2021. doi: 10.18653/v1/2021.emnlp-main.372.

Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Vishaal Udandarao, Ameya Prabhu, Adhiraj Ghosh, Yash Sharma, Philip H. S. Torr, Adel Bibi, Samuel Albanie, and Matthias Bethge. No "zero-shot" without exponential data: Pretraining concept frequency determines multimodal model performance. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Pavan Kumar Anasosalu Vasu, Hadi Pouransari, Fartash Faghri, Raviteja Vemulapalli, and Oncel Tuzel. MobileCLIP: Fast image-text models through multi-modal reinforced training. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15963–15974, 2024.

Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokula Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, and Hadi Pouransari. FastVLM: Efficient vision encoding for vision language models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19769–19780, 2025.

Pete Warden and Daniel Situnayake. *TinyML: Machine learning with TensorFlow Lite on Arduino and ultra-low-power microcontrollers*. O'Reilly Media, 2019.

Kan Wu, Houwen Peng, Zhenghong Zhou, Bin Xiao, Mengchen Liu, Lu Yuan, Hong Xuan, Michael Valenzuela, Xi Chen, Xinggang Wang, Hongyang Chao, and Han Hu. TinyCLIP: CLIP distillation via affinity mimicking and weight inheritance. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 21970–21980, 2023.

Chuanguang Yang, Zhulin An, Libo Huang, Junyu Bi, Xinqiang Yu, Han Yang, Boyu Diao, and Yongjun Xu. CLIP-KD: An empirical study of CLIP model distillation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15952–15962, 2024.

Kaicheng Yang, Tiancheng Gu, Xiang An, Haiqiang Jiang, Xiangzi Dai, Ziyong Feng, Weidong Cai, and Jiankang Deng. CLIP-CID: Efficient CLIP distillation via cluster-instance discrimination. In *AAAI Conference on Artificial Intelligence*, volume 39, pp. 21974–21982, 2025. doi: 10.1609/aaai.v39i20.35505.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive captioners are image-text foundation models. *Transactions on Machine Learning Research (TMLR)*, 2022.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 11975–11986, 2023.

Chaoyang Zhu and Long Chen. A survey on open-vocabulary detection and segmentation: Past, present, and future. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

Table 3: Training hyperparameters for TINYVLM.

Hyperparameter	**Value**
Optimizer	AdamW
Learning rate	1×10^{-3}
Weight decay	0.01
LR schedule	Cosine annealing
Warmup epochs	10
Total epochs	100
Batch size	32 per step; 256 per update (8-step grad. accum.)
Temperature τ	0.07
Distillation temp. τ_d	4.0
α_{distill}	0.5
α_{mat}	0.5
Queue size	4096
Matryoshka dims	{16, 32, 64, 128, 256}

Table 4: Dataset-specific prompt templates for text embedding computation.

Dataset	**Templates**
Default (all main results)	"a photo of a {}", "a picture of a {}", "an image of a {}", "a photo of the {}"
COCO (prompt study only)	the four default templates, plus "a photo of a {} in a scene", "a photo of a small {}", "a photo of a large {}"
LVIS	the four default templates, plus "a photo of one {}"
Core50	"a photo of a {}", "a photo of a {} on a table", "a close-up photo of a {}", "a photo of the small {}", "a household {}", "a {} object"
TiROD	"a photo of a {}", "a photo of a {} in a warehouse", "a {} on the ground", "an industrial photo of a {}", "a photo of a {} outdoors", "a robotic view of a {}"
Flowers102	"a photo of a {} flower", "a photo of a {}, a type of flower", "a close-up photo of a {}", "a photo of the {} flower", "a {} in a garden", "a bright photo of a {}"
Food101	"a photo of {}, a type of food", "a photo of a plate of {}", "a photo of {}", "a food photo of {}", "a close-up photo of {}", "{}, a type of cuisine"

A ADDITIONAL EXPERIMENTAL DETAILS

A.1 TRAINING HYPERPARAMETERS

Table 3 provides the complete training configuration for TINYVLM.

A.2 TEXT PROMPT TEMPLATES

All main results use the four **default** templates for every dataset, so that no dataset benefits from prompt tuning the others did not receive. The dataset-specific sets in Table 4 are used only where explicitly noted - the prompt-sensitivity study of Section C uses the 7-template COCO set. Class embeddings are the average over the applicable templates, L2-normalised, truncated to the target dimension, and re-normalised.

Algorithm 1 TinyVLM Training

Require: CLIP teacher f_T, student f_S, dataset $\mathcal{D}$, Matryoshka dims $\mathcal{M}$
Require: Hyperparameters: $\alpha_{\text{mat}}, \alpha_{\text{distill}}, \tau, \tau_d, T$
Initialize student weights; freeze teacher
for epoch $= 1$ to T **do**
  for each batch $\{(\mathbf{x}_i, \text{text}_i)\}_{i=1}^{N} \in \mathcal{D}$ **do**
    $\mathbf{e}_{\text{img}}^{T}, \mathbf{e}_{\text{txt}}^{T} \leftarrow f_T(\mathbf{x}, \text{text})$ *// Teacher (no grad)*
    $\mathbf{e}_{\text{img}}^{S} \leftarrow f_S(\mathbf{x})$ *// Student: vision only*
    $\tilde{\mathbf{e}}_{\text{txt}}^{(d)} \leftarrow \text{trunc}_d(\mathbf{e}_{\text{txt}}^{T})$ *// text side is the teacher's*
    $\mathcal{L}_{\text{cont}} \leftarrow \text{InfoNCE}(\mathbf{e}_{\text{img}}^{S}, \tilde{\mathbf{e}}_{\text{txt}}^{(d_{\max})}, \tau)$
    $\mathcal{L}_{\text{mat}} \leftarrow \sum_{d \in \mathcal{M} \setminus \{d_{\max}\}} w_d \cdot \text{InfoNCE}(\mathbf{e}_{\text{img}}^{S,[:d]}, \tilde{\mathbf{e}}_{\text{txt}}^{(d)}, \tau)$
    $\mathcal{L}_{\text{distill}} \leftarrow \tau_d^2 \, \text{KL}(\text{softmax}(\mathbf{S}^T/\tau_d) \| \text{softmax}(\mathbf{S}^S/\tau_d))$
    $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{cont}} + \alpha_{\text{mat}}\mathcal{L}_{\text{mat}} + \alpha_{\text{distill}}\mathcal{L}_{\text{distill}}$
    Update student via AdamW
  end for
end for
return Trained student f_S

A.3 MCU Deployment Details

Quantization. We use TensorFlow Lite full-integer post-training quantization: symmetric INT8 weights, and INT8 activations with per-tensor zero points (the deployed graph's output zero point is non-zero, and the firmware subtracts it before scoring). Text embeddings are quantized separately using symmetric INT8 with one scale per embedding *coordinate*, shared across classes (Section 3.3).

Memory Layout (STM32H7). On STM32H7, we use:

- Flash (2MB): firmware, the 936KB INT8 model, and the INT8 prototypes with their per-coordinate scales - 1,074KB in all (1,085KB at 256d), from the linker
- AXI SRAM (512KB): the TensorFlow Lite Micro tensor arena, a 512KB static buffer of which 464KB is used at peak
- DTCM (128KB): 1.2KB of static data, including the per-class prototype norms, and the stack; the heap is in D2 SRAM

Memory Layout (MAX78000). The MAX78000 uses dedicated CNN accelerator memory. Note that this platform runs the 4-layer accelerator-compatible network at 88×88 described in Table 14, not MobileNetV2-0.35×:

- CNN Weight Memory (442KB pool): 23.8KB backbone weights (5.4% utilization)
- CNN Data Memory (512KB pool): Intermediate activations
- Flash (512KB): 124.8KB total - firmware, 8.5KB projection weights, and 10.5KB INT8 class prototypes with their per-coordinate scales (24.4% utilization)
- SRAM (128KB): 8.8KB runtime variables

A.4 Training Algorithm

Algorithm 1 gives the full training procedure.

A.5 Memory Layout

Figure 3 shows where each part of the deployed model sits on the STM32H7.

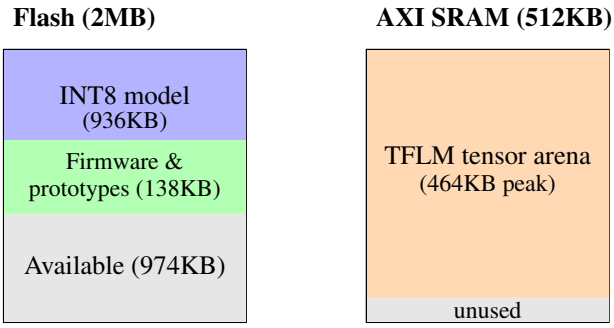

Figure 3: Memory layout for TINYVLM on STM32H7 at 128d (schematic, not to scale). Flash use (1,074KB) is from the linker, leaving 974KB free after the prototypes (984KB before them, the budget of Table 17); the arena peak is from running the same firmware under emulation (Section B.9). The arena is a 512KB static buffer. Static data (1.2KB, including the per-class prototype norms) and the stack sit in the 128KB DTCM, and the heap in D2 SRAM.

A.6 VLM SIZE COMPARISON

Table 5 places TINYVLM against the smallest published variant of each VLM family.

B ADDITIONAL ABLATION STUDIES

Table 6 gives the loss-component and backbone-width ablations summarised in Section 4.4.

Protocol note. The ablation tables and figures in this appendix were measured before the uniform-protocol rebuild of Table 1 and are internally consistent on that earlier protocol; they are not mixed with it. Two anchors let the two be read together: the student's COCO 256d figure is 27.7 here against 27.1 in Table 1, and the ViT-B/32 teacher's is 34.4 against 35.8. The differences come from the prompt protocol and, for the teacher, from an evaluation that scored only the 70 base classes on 4,801 images rather than all 80 on 4,952. Every comparison drawn in this appendix is within-protocol, but effect sizes, not only absolute values, can differ across protocols: widening the backbone from 0.25× to 0.50× gains $+1.2$ at 256d here and $+4.8$ in the uniform-protocol grid (Table 18), which was also trained with a multi-student harness whose warmup and numeric precision differ. We therefore read single-seed differences below twice the cross-seed standard deviation of the same dataset and dimension as unresolved throughout (population SD over three seeds; 1.2 points at 256d on COCO).

All ablations use MBV2-0.35× unless noted, 100 epochs on CC3M, single seed (42).

B.1 EMBEDDING QUANTIZATION

Table 7 gives prototype accuracy at each storage precision and dimension.

B.2 MATRYOSHKA DIMENSION WEIGHTING

Inverse weighting ($w_d \propto 1/d$) gives the highest 256-dim accuracy (28.7% vs. 27.8% equal), but the 0.9-point margin is a single seed against a cross-seed standard deviation of 0.6 at 256d (Table 1), so we keep equal weighting.

B.3 DATA SCALE

Accuracy scales sub-linearly with data: 500K samples (59% of full CC3M) already achieve 94% of full performance. A log-linear fit $a + b\log_{10}(N)$ to the three observations (100K→19.1, 500K→26.1, 854K→27.8 at 256d) gives $b \approx 9.5$ points per decade. Taken literally it would pass the ViT-B/32 teacher (34.4% in its earlier 70-class evaluation, 35.8% in Table 1) before 10M samples,

Table 5: Comparison of VLM approaches, each family represented by its *smallest published variant*. **Deployed memory is one convention throughout: FP32 weights, i.e. 4 bytes × the deployed-parameter column**, so every cell is derivable from its neighbour. Released checkpoints often ship at FP16/bfloat16, which halves these; we measured the released CLIP ViT-B/32 and ViT-L/14 archives at 338 and 890 MiB, and SmolVLM-256M's model card reports ≈0.8 GB of inference memory. TINYVLM's cell is instead STM32H7 flash from the linker (INT8 weights + firmware + prototypes), the strictest convention in the table. No claim turns on better than order-of-magnitude precision - the gap to a 1 MB budget is 1.5–4 orders. The distinction that matters for MCU deployment is the last two columns: dual encoders are counted as coupled deployments holding both towers; with class prototypes precomputed, as in standard CLIP zero-shot use, they would ship only their image tower, the quantity on the x-axis of Figure 1. **MCU viable**: fits 2 MB flash and 512 KB SRAM, the STM32H7 class that sets our target; the ESP32-S3 we measure (8 MB flash, PSRAM) would also hold LightCLIP's 2.45 MB INT8 image tower.

Method	Total params	Deployed params	Deployed memory	MCU viable
Full-scale VLMs				
CLIP ViT-B/32 (Radford et al., 2021)	151 M	151 M	605 MB	✗
CLIP ViT-L/14 (Radford et al., 2021)	428 M	428 M	1.71 GB	✗
OpenCLIP ViT-bigG/14 (Cherti et al., 2023)	≈2.5 B	≈2.5 B	≈10 GB	✗
Efficient dual-encoder VLMs				
TinyCLIP ViT-8M/16 (Wu et al., 2023)‡	11–23.4 M	11–23.4 M	44–94 MB	✗
MobileCLIP-S0 (Vasu et al., 2024)	53.8 M	53.8 M	215 MB	✗
LightCLIP + MBV2 (Nie et al., 2023)§	≈28–53 M	≈28–53 M	≈112–212 MB	✗
Small LLM-based VLMs				
SmolVLM-256M (Marafioti et al., 2025)	256 M	256 M	1.02 GB	✗
Ours (decoupled)				
TINYVLM†	**14.8 M**	**0.72 M**	**1,085 KB**	✓

‡TinyCLIP's name counts transformer bodies (8 M vision + 3 M text); the released checkpoint is 23.4 M once token embeddings are included - the same accounting ambiguity that affects our own row, which is why both totals are given. §LightCLIP reports only its image encoder (2.45 M, its §3.1) and no memory figure at all; the total here is ours, computed for the 8-layer text transformer it specifies at the width (512) of the DeCLIP codebase it is built on - ≈28 M excluding the 49,408-token embedding table, ≈53 M including it. Its 31.8% ImageNet is under YFCC15M-V2 pretraining, not a LAION/WIT-scale run. †Total includes the text tower's 12.9 M parameters, of which 12.8 M are the token embedding table, allocated but never trained or deployed (Section 3.1); deployed is the vision path. TINYVLM memory is total STM32H7 flash at 256d from the linker - firmware, INT8 weights, and class prototypes together - so it is an upper bound relative to the weights-only figures for the baselines.

Table 6: Ablation studies on COCO zero-shot accuracy (%). All variants use MBV2-0.35× unless noted, 100 epochs on CC3M. **Left:** loss component contributions. **Right:** backbone width vs. accuracy.

$\mathcal{L}_{\text{cont}}$	$\mathcal{L}_{\text{distill}}$	$\mathcal{L}_{\text{mat}}$	64d	128d	256d
✓			7.1	11.1	19.6
✓	✓		15.7	19.8	26.2
✓		✓	16.5	20.9	27.9
✓	✓	✓	16.8	20.9	27.8

Width	Params	64d	128d	256d
0.25×	238K	15.2	19.4	27.1
0.35×	396K	16.8	20.9	27.8
0.50×	688K	18.0	21.9	28.3

which a distilled student is not expected to do; three points spanning less than one decade cannot support an extrapolation across two, so we draw no conclusion about the data-limited regime beyond the sub-linear trend itself.

Table 7: Effect of text embedding quantization on COCO zero-shot accuracy (%) at different Matryoshka dimensions (80 classes), under per-coordinate scales with the dequantised prototypes re-normalised before the similarity argmax. Storage is lossless to INT4 at every dimension (INT4 sits within cross-seed noise of Float32) and costs 0.8 points at INT2/256d. An earlier version of this evaluation used per-row scales and omitted the re-normalisation, which lets uneven norm inflation act as a per-class bias and reported a spurious 23.20 at INT4/256d (-16.4%); the defect is largest at 256d because a single CLIP coordinate with $24\times$ the median coordinate amplitude enters the truncated prefix only at that dimension. KB counts the integer codes only; the d FP32 scales add 1 KB at 256d.

Precision	**64-dim**		**128-dim**		**256-dim**	
	Acc.	KB	Acc.	KB	Acc.	KB
Float32	16.82	20.0	20.92	40.0	27.75	80.0
Float16	16.82	10.0	20.94	20.0	27.79	40.0
Int8	16.78	5.0	20.92	10.0	27.77	20.0
Int4	16.94	2.5	20.84	5.0	27.89	10.0
Int2	15.73	1.2	19.39	2.5	26.96	5.0

Table 8: Matryoshka weighting ablation on COCO zero-shot accuracy (%).

Weighting	**64-dim**	**128-dim**	**256-dim**
Equal ($w_d = 1/4$)	16.8	20.9	27.8
Inverse ($w_d \propto 1/d$)	17.1	21.6	28.7
Log ($w_d \propto \log(256/d)$)	17.3	20.1	27.6

B.4 Teacher Model

On this earlier protocol ViT-L/14 (768-dim embeddings) is within 0.5 points of ViT-B/32 (512-dim) at 64d and 128d and trails at 256d (21.4 vs. 27.8). The teacher $\times$ width grid (Table 18), trained later with the published objective and hyperparameters in a multi-student harness, puts the stronger teachers slightly ahead at 128d instead - ViT-L/14 by 0.2 to 1.3 points and ViT-B/16 by up to 3.2 (0.1 behind at $0.50\times$) - and behind at 256d, where ViT-B/32 alone gains its coordinate 133: zeroing that coordinate removes $\approx$70% of the student's COCO gain over the $128 \to 256$ step (Section B.12). Either way a stronger teacher buys about one width step at 128d, consistent with the capacity argument: the student spans $\approx$93 PCA-95 dimensions against $\approx$277 for ViT-B/32 and $\approx$375 for ViT-L/14 (Figure 5).

B.5 Embedding Queue Size

The embedding queue stores the image and text embeddings of past steps, so each InfoNCE term scores its B positives against $B + Q$ candidates rather than B alone, where $B = 32$ is the per-step batch (gradient accumulation over 8 steps raises the optimizer batch to 256 but adds no negatives). We use $Q = 4{,}096$ in all experiments reported in this paper, providing $128\times$ more negatives than in-batch pairs. Table 11 sweeps $Q \in \{1024, 2048, 4096, 8192\}$ under identical protocol (seed 42, 100 epochs).

Table 9: Data scale ablation on COCO zero-shot accuracy (%).

Data	**64-dim**	**128-dim**	**256-dim**
CC3M 100K	10.2	14.6	19.1
CC3M 500K	14.1	19.1	26.1
CC3M Full (854K)	16.8	20.9	27.8

Table 10: Teacher model ablation on COCO zero-shot accuracy (%), earlier protocol, seed 42. Table 18 gives the later teacher × width grid.

Teacher	Embed Dim	64-dim	128-dim	256-dim
ViT-B/32	512	16.8	20.9	**27.8**
ViT-L/14	768	16.5	20.4	21.4

Table 11: Embedding queue size ablation on COCO zero-shot accuracy (%). Single seed (42), 100 epochs on CC3M. Pool size is $(B{+}Q)/B$ at the per-step batch $B{=}32$. ★: default configuration used in all other experiments.

Queue Q	Pool size	64-dim	128-dim	256-dim
1,024	33× batch	16.8	21.0	27.5
2,048	65× batch	17.2	21.6	27.9
4,096★	129× batch	16.8	20.9	27.8
8,192	257× batch	16.4	19.9	27.7

Across an 8× range of queue sizes, 256-dim COCO accuracy varies by only ±0.2 percentage points (27.5–27.9%). Lower Matryoshka dimensions show slightly more variation (±0.8 pp at 128d), but no monotonic trend emerges. This confirms the contrastive objective is robust to queue size in this regime, likely because even the smallest queue gives each 32-pair step 1,024 extra negatives, already ample for 80-class discrimination. Our default $Q = 4{,}096$ is a safe choice but not critical. $Q = 2{,}048$ marginally leads at 128d (21.6 vs. 20.9), but the differences are within single-seed noise and we retain $Q = 4{,}096$ as it was fixed before the sweep.

B.6 Alternative Backbone: ShuffleNetV2

The main paper explores three MobileNetV2 widths (0.25×/0.35×/0.50×). To assess whether our findings transfer to an architecturally different backbone at comparable parameter count, we train TinyVLM with a ShuffleNetV2 v0.5 vision backbone (342K parameters, comparable to MBV2-0.35× at 396K) under identical protocol (CC3M, 100 epochs, seed 42, ViT-B/32 teacher, equal Matryoshka weighting).

At comparable parameter count, ShuffleNetV2-0.5× matches MobileNetV2-0.35× within 1 percentage point at every Matryoshka dimension (28.7 vs. 27.8 at 256d), suggesting that zero-shot accuracy is determined primarily by backbone capacity rather than architecture family (the 0.9-point lead at 256d is a single seed, below our 1.2-point threshold). We deploy MobileNetV2 because our TFLite exporter and firmware support its operators; ShuffleNetV2's channel shuffle needs reshape, transpose, split and concatenation operators our firmware does not register, so its on-device cost is unmeasured.

B.7 Matryoshka vs. Naive Truncation

Figure 4 gives the accuracy-dimension trade-off referenced in Section 4.2. The naive-truncation baseline truncates CLIP ViT-B/32 embeddings without any Matryoshka training; PCA truncation projects the full CLIP embedding and therefore needs the 351 MB CLIP image encoder at inference.

Table 12: Alternative backbone ablation on COCO zero-shot accuracy (%) at comparable parameter count. Single seed (42), 100 epochs on CC3M.

Backbone	Params	64-dim	128-dim	256-dim
MobileNetV2-0.35×	396K	16.8	20.9	27.8
ShuffleNetV2-0.5×	342K	16.8	21.0	28.7

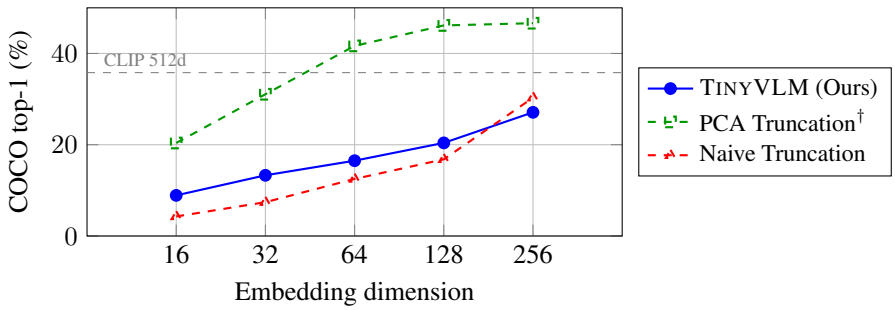

Figure 4: Zero-shot accuracy vs. embedding dimension on COCO. TinyVLM outperforms naive truncation of CLIP's own embedding at every dimension up to 128d (2.1× at 16d); at 256d, where truncated CLIP keeps half of its 512-d space, naive truncation scores 30.3 against 27.1. †PCA truncation of *full* CLIP embeddings is a reference, not a bound: the PCA is fitted on the evaluation images themselves and the prompts are a 6-template set, which is why it exceeds even the full 512-d teacher; it also needs the 351 MB CLIP image encoder at inference. Dashed line: CLIP full 512d (35.8%). The TinyVLM and naive series and the dashed line are on the protocol of Table 1.

Table 13: Prototype portability on COCO (top-1 %, 3-seed mean). The student is held fixed; only the text encoder that produces the class prototypes changes. Every encoder other than the distillation teacher performs at chance.

Prototype source	**16d**	**32d**	**64d**	**128d**	**256d**
CLIP ViT-B/32 *(distillation teacher)*	**8.9**	**13.3**	**16.5**	**20.4**	**27.1**
CLIP ViT-L/14	0.8	1.5	1.2	1.1	0.6
OpenCLIP ViT-B-32 (LAION-2B)†	1.8	1.1	1.3	1.4	1.4
OpenCLIP ViT-B-16 (LAION-2B)	0.5	0.7	1.8	1.1	1.3
Random chance	1.3	1.3	1.3	1.3	1.3

†Same architecture and embedding width as the teacher; differs in training data and recipe.

B.8 Prototype Portability Across Dimensions

Table 13 gives the 3-seed, per-dimension version of the 'raw' column of Table 2 (which is seed 42): the collapse to chance holds at every Matryoshka dimension, not only at 256d.

B.9 MCU Deployment Benchmarks

Table 14 gives the per-platform figures summarised in Section 4.3.

Table 15 gives accuracy at the resolution the boards run. Every other accuracy in the paper is the float32 student at 224×224; the rows below vary resolution and network precision one at a time on the same images.

B.10 Accuracy vs. Vocabulary Size

Table 16 gives the full operating-point sweep summarised in Section 4.5. For each vocabulary size K we draw 20 random subsets of the 80 COCO categories, restrict evaluation to images whose ground-truth class falls in the subset, and classify among those K prototypes. Values are means over subsets and over 3 seeds; top-5 at $K{=}5$ is trivially 100%.

B.11 Class Prototype Storage Cost

Because the vision encoder is vocabulary-independent, the only deployment cost that grows with the class list is the stored prototype table: exactly $K \cdot d \cdot 1$ bytes at INT8, plus d FP32 scales shared

Table 14: MCU deployment benchmarks for TINYVLM. ESP32-S3 and MAX78000 are measured on hardware via UART. STM32H7 and ESP32-S3 run the MBV2-0.35× INT8 encoder at 128×128 input; MAX78000 runs a smaller accelerator-compatible network at 88×88 (see †). SRAM is the TFLite Micro arena in use (MAX78000, which does not run TFLite Micro: static SRAM from the linker).

Platform	Freq. (MHz)	Flash (KB)	SRAM (KB)	Latency (ms)	Energy (mJ)	FPS
STM32H7[§]	480	1,074	464	—	—	—
MAX78000[†]	100	125	9	2.3	0.10[*]	430
ESP32-S3[‡]	240	1,288	464	396.7	49.8	2.5

[§]Flash from the linker. SRAM is the arena reported by the same binary running under emulation (Renode, STM32H753 model), which reproduces the board's earlier hardware-measured arena to the byte; the board failed before latency could be measured on this graph. [†]4-layer accelerator-compatible CNN (3→8→16→32→64) at 88×88 with a 64-dim embedding, scored against the first 64 coordinates of its stored 128-dim prototypes, *not* MobileNetV2-0.35×; latency is full-pipeline (861 μs accelerator + 1,460 μs on the Cortex-M4 for pooling, projection and prototype scoring). Carries none of the accuracy results reported elsewhere. [*]Whole board, measured with a Nordic PPK2 sourcing 5.0 V into the board's USB-power input with USB disconnected: 43.9 mW while inferring back to back, 0.10 mJ per inference including the power-management IC and the on-board debug MCU (the rated 200 mW would give 0.46 mJ). The chip's own rail is not exposed on this board, and no idle state is a usable baseline - with the CPU asleep the board draws 54.1 mW, more than while inferring - so unlike the ESP32-S3 figure this is not an above-idle increment. [‡]Arena in external PSRAM; flash is the single application image (firmware + model + prototypes). Energy measured with a Nordic PPK2 sourcing 3.3 V to the module: inference draws 125.4 mW above a 130.3 mW idle, 49.8 mJ per inference (101.5 mJ whole-module); the rated 181.5 mW is 45% above the inference increment and 29% below whole-module draw. All rows use 128-dim prototypes; the 256-dim configuration of Table 1 adds 10.5 KB (INT8 table and scales).

Table 15: COCO top-1 (%) at the deployed resolution. Seed-42 grid-control checkpoint (Table 18; the deployed model) on 2,047 of the 4,952 labelled COCO val2017 images, 80 classes unless marked, all scored against the deployed per-coordinate INT8 prototypes. The INT8 row runs the deployed TensorFlow Lite model through the reference kernels, which are bit-exact with the boards. Under the paper's protocol (float prototypes, 224×224) the same images give 28.1% at 256d and 48.8% at K=10. Input resolution, not quantisation, accounts for the gap.

Network	Input	128d	256d	256d, K=10
Float32	224×224	22.2	27.9	—
Float32	192×192	20.0	25.6	—
Float32	160×160	16.9	21.6	—
Float32	128×128	11.8	16.8	—
INT8 (deployed)	128×128	11.0	16.0	37.9
INT8, coord. 133 masked	128×128	—	12.2	37.2

across classes (1.0 KB at 256d, independent of K). Table 17 gives this cost across vocabulary sizes and Matryoshka dimensions, against the ≈984 KB of STM32H7 flash left free once firmware and INT8 weights are placed.

B.12 THE ROGUE TEACHER COORDINATE

Paragraph (ii) of Section 4.5 attributes the non-monotonic prefix curves on fine-grained data to CLIP ViT-B/32 coordinate 133 rather than to accumulated tail noise. This appendix gives the evidence and, more importantly, the corrections that do *not* work.

Coordinate 133 carries ≈24× the median coordinate amplitude (≈560× in energy). Because $128 < 133 \leq 256$ it is absent from every Matryoshka prefix except the largest. Zeroing it (3-seed means,

Table 16: COCO accuracy (%) as a function of deployed vocabulary size, at each Matryoshka dimension. Float32 student at 224×224; 3-seed mean, 20 random class subsets per size, drawn independently at each dimension; no spread is reported. The deployed INT8 pipeline at 128×128 is in Table 15.

	Vocabulary size K				
Metric / dim	**5**	**10**	**20**	**40**	**80**
Top-1, 256d	65.3	48.9	46.6	33.6	27.1
Top-1, 128d	59.7	48.1	34.4	27.5	20.4
Top-1, 64d	54.7	38.3	34.0	22.9	16.5
Top-1, 32d	51.7	37.8	25.6	20.0	13.3
Top-1, 16d	39.2	31.9	16.3	14.0	8.9
Top-3, 256d	90.7	76.4	70.4	54.1	45.1
Top-5, 256d	—	88.0	80.4	64.0	55.4

Table 17: Class prototype storage (KB, INT8) as a function of vocabulary size K and embedding dimension d, against the ≈984 KB of STM32H7 flash remaining after firmware and model weights. Matryoshka truncation is what makes the large-K regime feasible: LVIS needs 301 KB at 256d but only 38 KB at 32d.

Vocabulary	**16d**	**32d**	**64d**	**128d**	**256d**
K=10 (Core50)	0.2	0.3	0.6	1.2	2.5
K=80 (COCO)	1.2	2.5	5.0	10.0	20.0
K=101 (Food101)	1.6	3.2	6.3	12.6	25.2
K=1,203 (LVIS)	18.8	37.6	75.2	150.4	300.8
K=5,000	78.1	156.2	312.5	625.0	1250.0*

*Only this entry exceeds the remaining flash. Every other configuration fits, including a 5,000-class vocabulary at 128d. Prototypes are read from flash in place; the only startup cost is computing their K dequantised norms. The per-inference cost is the similarity loop, measured at 790 μs for K=80 at 128d on ESP32-S3 and scaling as $O(Kd)$ - 0.2% of the 396.7 ms inference, so even K=5,000 at 128d adds only ≈49 ms.

scored with the dataset-specific prompts of Section A.2: 7 templates for COCO) moves both datasets at once:

	128d	**256d**	**128 → 256**	**256d w/o 133**	**128 → 256 w/o it**
Food101	5.15	3.56	−1.59	5.50	+0.35
COCO	20.49	27.93	+7.44	22.73	+2.24

No rescoring repairs the table. The non-leaky corrections below were each tested at 3 seeds. Every one costs per-image COCO 5–8 points at 256d - the "person" points of Section B.15 - and only rank-1 removal of the prototype direction is monotone on all three datasets. Under the class-balanced headline the sign flips: at seed 42, rank-1 removal lifts COCO from 21.2 to 22.4 and ImageNetV2 from 6.3 to 7.2, and zeroing coordinate 133 to 22.2 and 7.4 (Table 22).

Variant (256d)	**COCO**	**Food101**	**Flowers102**	**monotone?**
unmodified (dataset-specific prompts)	**27.93**	3.56	5.25	no
rank-1 removal (prototype direction)	21.55	5.97	7.07	**yes**
drop coord 133	22.73	5.50	6.42	no
drop 133+92+121	21.10	5.74	6.50	no
per-coordinate z-score (image stats)	19.86	5.53	6.57	no
per-coordinate z-score (per-modality)	20.80	**6.60**	**7.81**	no

Table 18: Absolute COCO top-1 (%) of the nine grid students at 128d and 256d (seed 42, full validation set, 100 epochs on CC3M). The ViT-B/32 × 0.35 cell is the control: 27.12 against 27.8 for the original seed-42 run.

	0.25×		**0.35×**		**0.50×**	
Teacher	128d	256d	128d	256d	128d	256d
CLIP ViT-B/32	18.62	25.26	21.14	27.12	23.10	30.05
CLIP ViT-L/14	19.59	20.36	22.48	23.08	23.26	23.83
CLIP ViT-B/16	21.81	21.18	23.57	21.87	23.02	22.64

Per-coordinate standardisation is what Timkey & van Schijndel (2021) prescribe for rogue dimensions; it neither restores monotonicity nor protects per-image COCO. Scored class-balanced, the rescored table would be the better one; we report both rather than change the paper's protocol at submission.

Changing the teacher, not the embedding. The zeroing experiment above removes the coordinate from a fixed teacher. The complementary test is to change the teacher so the coordinate cannot reach the prefix at all. CLIP ViT-B/16 places its dominant coordinate at index 362 and ViT-L/14 at 316, both outside a 256-d prefix. We trained a 3-teacher × 3-width grid (0.25/0.35/0.50×) for 100 epochs on CC3M with the published objective and hyperparameters in a multi-student harness (warmup and precision differ), one seed, all widths for a teacher trained in a single process so they share batches and teacher outputs exactly. The ViT-B/32 × 0.35× cell is a reproduction control and returns **27.12**% on COCO at 256d against 27.8 for the original seed-42 run (27.1 is the 3-seed mean). This grid-control checkpoint is the model deployed on hardware (Section 4.3); the three seeds of Table 1 come from the single-student harness.

Teacher	**Dominant coord.**	**COCO**	**Food101**	**Flowers102**
		128 → 256 step at widths 0.25/0.35/0.50×		
CLIP ViT-B/32	133 *(inside)*	+6.64/+5.98/+6.95	−0.63/−0.72/−0.64	−0.05/+0.06/+0.08
CLIP ViT-L/14	316 (outside)	+0.77/+0.60/+0.57	+0.67/+0.40/+0.80	+2.46/+2.05/+1.39
CLIP ViT-B/16	362 (outside)	−0.63/−1.70/−0.38	+0.83/+0.93/+1.14	+1.48/+2.57/+1.84

On all three datasets the inside-prefix teacher sits apart from the two outside-prefix teachers, with gaps of 5.21, 1.03 and 1.31 points. Against twice each dataset's own cross-seed standard deviation at 256d (population SD over three seeds: 1.2, 0.8 and 1.2 points for COCO, Food101 and Flowers102), COCO clears by far and the two fine-grained gaps only marginally; these are single-seed cells, so we treat the fine-grained separations as suggestive rather than resolved. The direction differs by dataset because the coordinate is the prototypes' shared direction (Table 22): it raises per-image COCO by around six points, all on "person", and depresses Food101, and under a teacher that cannot reach the prefix both effects vanish together. These grid gaps are per image; the nine grid students have not been re-scored class-balanced. At matched width 0.35×, COCO's 128 → 256 step is +5.98 for ViT-B/32, +0.60 for ViT-L/14 and −1.70 for ViT-B/16, so most of the apparent per-image dimension-scaling benefit on COCO - ≈70% by zeroing, nearly all by this comparison - is attributable to one coordinate of *the teacher*, not to the student using its extra capacity. This is the experiment that could have falsified the account in paragraph (ii) of Section 4.5; it did not.

Scope. The zeroing and rescoring numbers above are one teacher (ViT-B/32) and one student width (0.35×), scored with the dataset-specific prompts of Section A.2; Core50 and LVIS were not re-run. The 128 → 256 jump is specific to this teacher: the teacher × width grid above shows it only for the inside-prefix teacher, while ViT-B/16's COCO curve instead peaks at 128d at every width (Table 18).

Table 19: CC3M pretraining data coverage per evaluation dataset (LVIS not computed). Coverage = fraction of 854K captions containing a class name (case-insensitive word match), which overcounts short polysemous names such as Core50's "can", "glass" and "ball".

Dataset	Classes	Mean %	Median %	Min %	Max %
COCO	80	0.23	0.05	0.000	7.30
Core50	10	0.35	0.09	0.000	1.18
TiROD	13	0.14	0.02	0.000	0.80
Flowers102	102	0.00	0.00	0.000	0.06
Food101	101	0.00	0.00	0.000	0.09

B.13 Pretraining Data Coverage Analysis

Table 19 quantifies how frequently each evaluation dataset's class names appear in the 854K CC3M training captions (case-insensitive word-boundary match). COCO classes appear in 0.23% of captions on average versus 0.002% for Flowers102 - a 100× gap that tracks the teacher-retention disparity (76% vs. 9%). Core50's mean is driven by short polysemous names ("can", "glass", "ball") whose word matches need not refer to the object, so it is evidence neither way. Low coverage and fine granularity coincide in the other datasets, so their effects cannot be separated, and we ran no control that holds coverage fixed.

B.14 CLIP-Pseudo-label Baseline Details

The supervised baseline reported in Section 4.2 uses the same MobileNetV2-0.35× backbone and 256-dim projection head as TinyVLM, but replaces all contrastive/distillation losses with a single 80-way softmax cross-entropy on CLIP-ViT-B/32 pseudo-labels. Pseudo-labels are produced by running the teacher over CC3M with a 6-template COCO prompt ensemble (logit scale $= 100$) and recording the argmax class for each image. We keep the text-encoder pathway of TinyVLM frozen, so the baseline trains the same 724 K backbone-plus-projection path as TinyVLM together with a 20 K classification head. The two are therefore matched on trained parameters to within 3%. Training uses 100 epochs of AdamW (lr$= 10^{-3}$, weight decay 0.01) with a 10-epoch linear warmup and cosine decay. Evaluation is standard COCO validation top-1 via the 80-way classification head (no text encoder at inference).

B.15 Evaluation Protocol Notes

Two details of the evaluation code matter, and every number in this paper uses the form described here.

TiROD class order. Class prototypes and dataset labels must be enumerated in the same order. The loader enumerates `base_categories` followed by `novel_categories`, and prototypes are built from that same list rather than from the module-level category list, which agrees on the first six of thirteen classes and differs on the rest. The evaluators assert the two orders match before scoring, since a mismatch is silent: the run completes and returns a plausible number.

Which projection is scored. The architecture contains per-dimension Matryoshka MLP heads, but no term in the training objective references them, so they receive no gradient and remain at initialisation. All reported accuracies score the deployed path instead - `encode_image` followed by prefix truncation and re-normalisation - which is the path the MCU firmware runs and is identical across every dataset in Table 1. All six student columns there come from one script, `eval_embedding_space.py -teacher_only`, which scores this path against 4-template teacher prototypes; the teacher row uses `eval_clip_teacher.py` with the same prototypes. An ImageNet-1K evaluation that scored those heads measured a random projection; the next paragraph re-scores ImageNet's classes on the deployed path.

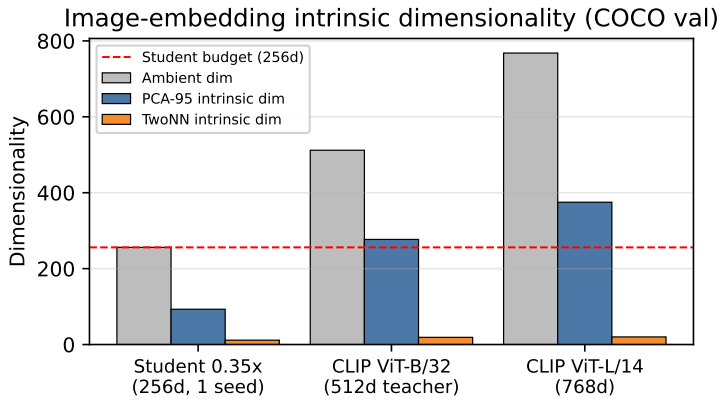

Figure 5: **Intrinsic dimensionality of image embeddings on 2K COCO val images.** Bars show ambient dimension, PCA-95 (smallest k explaining 95% of variance), and the TwoNN estimator (Facco et al., 2017). The Matryoshka objective front-loads the student's variance into early coordinates, which may lower its PCA-95 count relative to the teachers'. The trained 0.35× student spans only ≈93 effective dimensions. CLIP ViT-B/32 requires ≈277 dimensions - just above the 256d student budget - whereas CLIP ViT-L/14 requires ≈375, exceeding the budget by 1.5× and empirically consistent with the capacity reading of Section 4.4: the student cannot span the stronger teacher's extra geometry, so that teacher adds little.

Core50 prototype strings. The ten Core50 prototypes are built from "plug adapter", "mobile phone", "scissors", "light bulb", "can", "glass", "ball", "marker", "cup" and "remote control". Core50's sixth category is eyewear ("glasses"), so that class was scored against a drinkware name for teacher and student alike; we have not re-scored it.

The deployed pipeline with coordinate 133 masked. Zeroing column 133 of the 256-d INT8 prototype table and re-normalising its rows is equivalent, for the argmax, to masking both sides, so it needs a new table and nothing else on the device. On the 2,047 images of Table 15 it moves class-balanced top-1 from 12.9 to 13.5 and "person" predictions from 8.9% to 3.0% of images, and lowers per-image top-1 from 16.0 to 12.2 and $K{=}10$ from 37.9 to 37.2 (`scripts/diagnostics/deployed_no133.py`).

ImageNet classes on the deployed path. We score the matched-frequency split of ImageNetV2 (Recht et al., 2019) - 10 images for each of the 1,000 ImageNet-1K classes, so top-1 is already class-balanced and chance is 0.1% - with CLIP's ImageNet class names, the 4-template protocol of Table 1, the same 224 px input for teacher and student, and the deployed grid-control checkpoint (seed 42; `scripts/diagnostics/eval_imagenetv2_zeroshot.py`). The student reaches 6.3% top-1 at 256d and 7.0% at 128d, 70× chance but 12% and 13% of the teacher's 52.6% (Table 20): retention between COCO's and the fine-grained sets'. We have no coverage or granularity measure for ImageNet's names that would place it there independently. As on Food101, accuracy falls from 128d to 256d, and the predictions concentrate there: the five most-predicted classes take 9.7% of all predictions at 128d and 19.2% at 256d (0.5% if uniform; 1.5% for the teacher), and 443 and 591 of the 1,000 classes are never predicted. That is the direction coordinate 133 pushes the fine-grained sets (Section 4.5(ii)), and zeroing it as in Section B.12 tests the account on a set it was not fitted to: at 256d top-1 rises from 6.3% to 7.4% (top-5 16.7 to 20.7), above the 128d score, and the concentration halves (top-5 share 10.0%, 460 classes never predicted). Coordinate 133 carries 76% of the squared norm of the mean 256-d prototype, so it is the common direction every prototype shares. CLIP's 80 ImageNet templates (Radford et al., 2021) raise the teacher to 53.4% and move the student by at most 0.5 at any dimension (under 0.1 at 128d and 256d), so the gap is not a prompt artefact. Nor is it the input transform. The paper resizes images to 224×224 without cropping, as the student was trained; under CLIP's reference transform (short side to 224, then a centre crop) the teacher rises from 52.6% to 55.4% while the student moves by at most 0.4 (6.4% at 256d, 7.1% at 128d), so retention is 12% either way. The squash costs the teacher 2.8

points here but at most 0.5 on COCO (Section B.15). With the reference transform and CLIP's 80 templates, the protocol of CLIP's own ImageNet evaluation, the teacher reaches 55.9% and the student 6.7% at 256d, again 12%. ImageNetV2 is a harder re-collection of ImageNet's validation set (Recht et al., 2019), so these are not ImageNet-1K validation numbers, and they are one seed.

Table 20: Zero-shot accuracy (%) on ImageNetV2 (matched frequency; 10,000 images, 1,000 classes; chance 0.1 top-1, 0.5 top-5), deployed checkpoint (grid control, seed 42), 4-template protocol and 224×224 resize unless noted. Top-5 share: the fraction of all predictions taken by the five most-predicted classes (0.5 if uniform). Never predicted: classes that no image is assigned to.

	16d	32d	64d	128d	256d	Teacher (512d)
Top-1	1.9	4.1	5.7	7.0	6.3	52.6
Top-5	5.5	10.6	15.6	18.7	16.7	80.0
Top-1, 80 templates	2.1	4.5	5.9	7.0	6.3	53.4
Top-5 share	—	—	—	9.7	19.2	1.5
Never predicted	—	—	—	443	591	3
Top-1, coord. 133 zeroed	—	—	—	—	7.4	—
Top-1, centre crop	1.6	4.1	6.0	7.1	6.4	55.4
Top-1, centre crop, 80 templates	2.1	4.8	6.3	7.1	6.7	55.9

COCO's "person" class, class-balanced accuracy, and the input transform. Table 21 scores the deployed checkpoint on all 4,952 labelled COCO val2017 images under the protocol of Table 1 (`scripts/diagnostics/coco_person_hub_and_crop.py`); it reproduces that table's teacher (35.8) and the 67-class figures of Section 4.2. Three readings. First, the student's "person" predictions are informative: it predicts the class for 16.8% of images and is right 57% of the time, against a 31.6% base rate, while the teacher predicts it for 1.8% (its prototype loses to the more specific objects in crowded scenes). Second, class-balanced accuracy over all 80 classes (21.2) is below the 67-class figure (23.6) because the 13 classes with fewer than ten images average 8.5% for the student and 59.5% for the teacher; we therefore use all 80 for the headline. Third, the per-image gain from 128d to 256d is the "person" class: its points rise from 3.4 to 9.6 while class-balanced accuracy falls slightly. LVIS is scored on its v1 val split: 19,626 images with at least one annotation (all present on disk, as the loader requires), each labelled with its most-annotated category; 740 of its 1,203 categories label some image, 281 label at least ten, and the most frequent label, "zebra", covers 1.4% (`scripts/diagnostics/lvis_label_hist.py`). Chance is 1/1,203 for a uniform guess over its prototypes and 1/740 over the classes that occur; both round to the 0.1 of Table 1. The paper resizes every image to 224×224 without cropping, as in training; CLIP's reference transform (short side to 224, centre crop) moves the teacher by at most 0.5 and the student by at most 0.9 on any column, and class-balanced retention by under one point.

Table 21: COCO val2017 (4,952 images), deployed checkpoint (seed 42), 4-template prototypes, under the paper's 224×224 resize and CLIP's reference short-side resize with centre crop. Macro: mean per-class accuracy over all 80 classes, and over the 67 with ≥10 images. "person": share of all images predicted "person", the precision of those predictions (base rate 31.6%), and the per-image points they contribute.

Input	Model	Per image	Macro-80	Macro-67	"person" pred.	Precision	Points
Resize	Student 256d	27.2	21.2	23.6	16.8	57.2	9.6
	Student 128d	21.2	21.8	24.2	6.9	49.0	3.4
	Teacher	35.8	54.9	54.1	1.8	43.7	0.8
Crop	Student 256d	27.8	21.3	24.3	16.5	57.6	9.5
	Student 128d	22.1	22.0	24.7	7.1	49.0	3.5
	Teacher	36.0	55.4	54.5	1.6	43.2	0.7

The shared prototype direction. Table 22 tests whether coordinate 133 is the prototypes' mean direction (`scripts/diagnostics/coord133_mean_direction.py`). It carries 79% of the squared norm of the mean 256-d COCO prototype (76% on ImageNetV2). Zeroing it moves 18% of COCO predictions: "person" loses 8.3 points of prediction share and "book" 4.3, spread

over "bench", "dining table", "sports ball" and others; on ImageNetV2 the classes that lose are generic ones ("website" 5.2, "pig" 3.1, "stage" 2.9). Projecting the mean direction out of images and prototypes behaves like zeroing and lifts class-balanced 256d above 128d on both sets; for the teacher it is neutral on COCO and costs 1.5 on ImageNetV2. Centring the prototypes alone, with re-normalisation, does the opposite: "person" takes a quarter of COCO's predictions and class-balanced accuracy falls.

Table 22: The shared prototype direction, deployed checkpoint (seed 42), 4-template prototypes, 224×224 resize. Full: the paper's scoring. No 133: coordinate 133 zeroed in images and prototypes, re-normalised. Projected: the mean-prototype direction removed from images and prototypes. Centred: prototypes minus their mean, re-normalised, images unchanged. COCO per image and class-balanced (80 classes), the share of COCO predictions that are "person", and ImageNetV2 top-1 (class-balanced by construction).

	COCO per image	COCO class-bal.	"person" share	ImageNetV2
Student 128d, full	21.2	21.8	6.9	7.0
Student 256d, full	27.2	21.2	16.8	6.3
Student 256d, no 133	22.6	22.2	8.5	7.4
Student 256d, projected	22.1	22.4	7.8	7.2
Student 256d, centred	30.5	19.5	24.2	4.7
Teacher 512d, full	35.8	54.9	1.8	52.6
Teacher 512d, projected	35.8	55.3	1.5	51.0

B.16 Three Diagnostics: Hubness, Effective Dimension, and Within-Dataset Coverage

All three are computed from the deployed checkpoint (the grid control, ViT-B/32 × 0.35×, seed 42) at 256d; scripts in `scripts/diagnostics/`.

Prototype hubness. Counting how often each prototype is predicted (its k-occurrence) tests whether a few prototypes absorb the predictions.

	classes	**k-occ. skew**	**never predicted**	**top-5 share**	**uniform**
COCO	80	+4.26	13	40.9%	6.2%
Core50	10	+1.52	1	**97.9%**	50.0%

On Core50 five of ten prototypes absorb 98% of all predictions, so the student is not choosing among ten classes at all - which is what a near-chance score looks like from the inside. COCO is skewed too, with 13 of 80 prototypes never predicted, but the mass is spread far more evenly relative to the uniform share. Section 4.5(iii) cites this measurement; we do not claim hubness as the cause, since a collapsed prediction distribution is equally consistent with the embedding being uninformative on this domain.

Effective dimension. PCA-95 on the same embeddings gives 94 dimensions on COCO, consistent with the ≈93 used in the capacity argument, and 38 on Core50. The Core50 figure should not be read as a smaller student: it is the same encoder, and a narrower image distribution yields fewer principal directions.

Coverage within one dataset. The coverage claim in Section 4.5(i) rests on a comparison *between* datasets, which confounds coverage with granularity. Testing it *within* COCO holds granularity roughly fixed - 80 classes, one protocol, one model. Over the 66 classes with at least 10 validation images and a non-zero teacher score, Spearman correlations are +0.38 between CC3M coverage and student accuracy and +0.52 between coverage and teacher retention; teacher accuracy and student accuracy correlate at +0.65. The two COCO classes absent from CC3M average 5.2% student accuracy. Coverage therefore tracks retention within a dataset as well as across datasets, which is the weaker claim the paper makes.

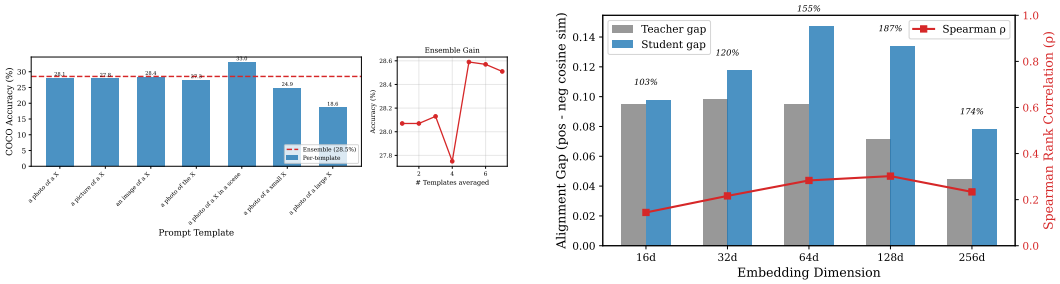

Figure 6: **Left:** per-template zero-shot accuracy on COCO at 256d, with the 7-template ensemble as a dashed line: accuracy varies widely across templates, and the ensemble exceeds the per-template mean. **Right:** cross-modal alignment fidelity across Matryoshka dimensions: the student *amplifies* the teacher's positive-negative alignment gap (bars) while only partially preserving rank order (Spearman ρ, right axis). Both panels show the seed-42 run; the 3-seed means are given in the text of this section.

Table 23: Retrieval metrics on COCO val2017 (4,952 images, 80 classes). I2T R@1 is zero-shot top-1 by construction; it differs from Table 1 only through the 7-template prompts. T2I R@K is, for each class, the fraction of that class's images ranked in the top K, averaged over classes; a class with n images cannot exceed K/n (about 1.6% at K=1 for the average class of 62 images). I2T R@1 and R@5 are 3-seed means.

	I2T (Image → Text)				**T2I** (Text → Image)			
Dim	R@1	R@5	R@10	MRR	R@1	R@5	R@10	MRR
64	16.7	33.6	44.5	26.4	0.7	2.9	4.9	2.4
128	20.5	43.0	56.1	32.2	0.9	3.3	5.8	2.8
256	**27.9**	**56.7**	**70.4**	**41.5**	0.5	2.3	4.0	2.0

C VISION-LANGUAGE ALIGNMENT CHECKS

Since TINYVLM deploys precomputed text embeddings at inference, we verify that the learned alignment is genuine rather than a memorized label set. Four checks on COCO agree. **Prompt ensembling:** per-template accuracy spans 18.6–32.9% (σ=4.0) at 256d, and ensembling 7 templates yields 27.94% against a per-template mean of 26.37%; the +1.57 pp ensemble gain is consistent with cross-modal structure rather than single-template matching, though the best single template (32.9%) beats the ensemble and the spread shows that accuracy is prompt-sensitive. **Alignment fidelity:** the student *amplifies* teacher alignment geometry, with preservation ratios 1.54/1.87/1.72× at 64/128/256d, while Spearman $\rho \approx 0.23$–0.30 means the student does *not* preserve the teacher's ordering of classes. **Retrieval:** I2T R@1 is zero-shot top-1 by construction (27.9 with the 7-template COCO prompts, against 27.1 in Table 1), and R@5=56.7%, R@10=70.4% at 256d improve monotonically with dimension. **Synonym robustness:** replacing all 20 evaluated COCO class names with their most dissimilar synonyms retains 76.6% of canonical accuracy (21.4% vs. 27.9%, per-class $\sigma < 2$), consistent with semantic rather than lexical matching.

The remainder of this appendix gives the supporting protocols and per-class breakdowns. The prompt, alignment, synonym and I2T R@1/R@5 numbers in the text are 3-seed means (seeds 42, 123, 456); the per-template range and the other columns of Table 23 were not recorded per seed, and the figures show the seed-42 run. All are evaluated on COCO val2017 (4,952 images, 80 classes).

Prompt templates used. The 7-template ensemble used in Figure 6 is the COCO-specific set listed in Section A. Per-template accuracy at 256d ranges from 18.6% ("a photo of a large {}") to 32.9% ("a photo of a {} in a scene"), with mean 26.37% and ensemble 27.94%.

Cross-modal fidelity protocol. For each image we compute positive alignment $\cos(\mathbf{v}, \mathbf{t}_y)$ and mean negative alignment $\frac{1}{K-1}\sum_{j \neq y} \cos(\mathbf{v}, \mathbf{t}_j)$, using teacher text embeddings $\mathbf{t}$ (averaged over the

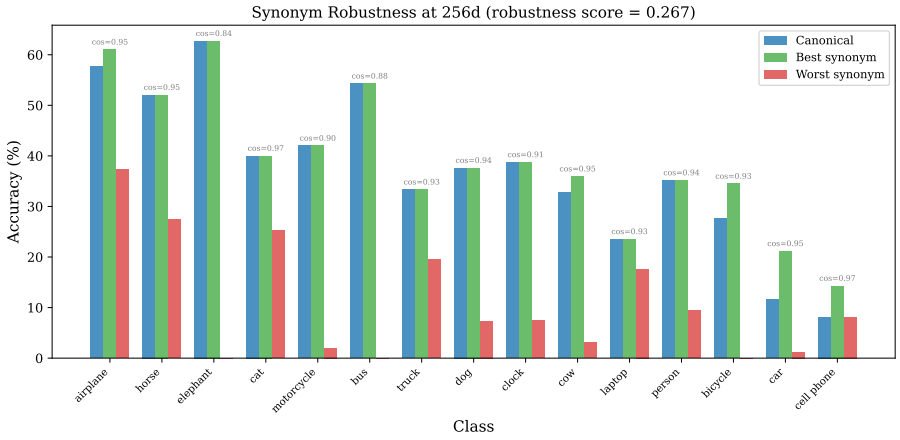

Figure 7: Synonym robustness on COCO at 256 dimensions (seed-42 run): canonical, best-synonym and worst-synonym accuracy for the 15 of the 20 evaluated classes with the highest mean accuracy across synonyms (the plot is capped at 15 for readability). The robustness score in the title is the class-averaged ratio of worst-synonym to canonical accuracy. Replacing all 20 names at once with their most dissimilar synonyms retains 76.6% of canonical accuracy (text), indicating TINYVLM captures semantic meaning rather than memorizing lexical surface forms.

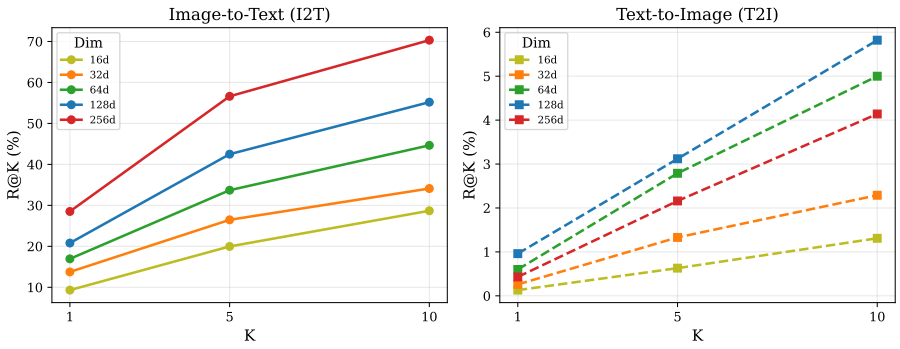

Figure 8: Image-to-text (I2T) retrieval curves on COCO across Matryoshka dimensions. R@K increases monotonically with K, and higher-dim embeddings dominate at every rank, confirming the Matryoshka hierarchy is effective for retrieval as well as classification. R@1 is close to the zero-shot accuracy of Table 1 (27.9 vs. 27.1 at 256d; 7-template prompts).

7-template ensemble) as a common reference. The alignment gap is the difference, and *preservation* is the ratio of the student's gap to the teacher's. Spearman ρ is computed per image over the 80-dimensional similarity vector and averaged.

Synonym set. For the synonym study we selected 20 COCO classes with ≥ 30 validation samples and clear lexical synonymy (e.g., car/automobile/vehicle/sedan; dog/canine/puppy/hound). Each class has 3–5 synonyms; canonical accuracy, best-synonym accuracy, and worst-synonym accuracy are reported per class in Figure 7. The full-substitution worst-case run replaces all 20 canonical class names with their most dissimilar synonym simultaneously.

D ADDITIONAL RESULTS

D.1 PER-CLASS ZERO-SHOT ACCURACY

Figure 9 shows the per-class zero-shot accuracy on COCO compared to the CLIP teacher.

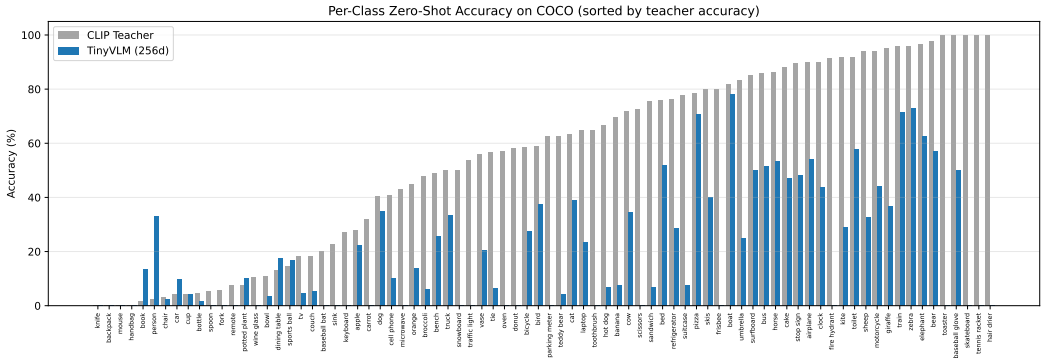

Figure 9: Per-class zero-shot accuracy comparison between TINYVLM (256-dim) and CLIP teacher (512-dim) on COCO val. Classes are sorted by teacher accuracy.

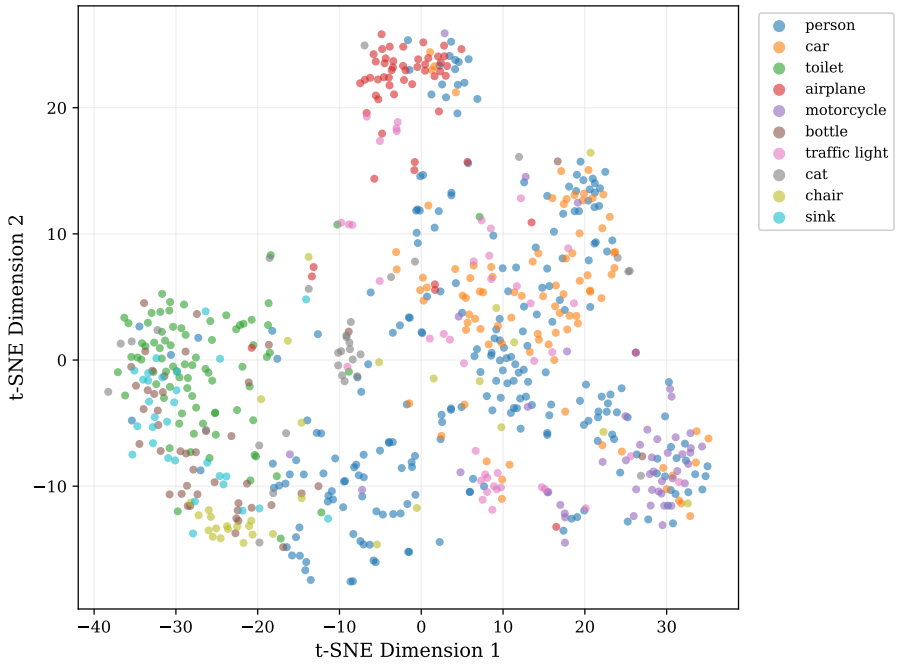

Figure 10: t-SNE visualization of TINYVLM image embeddings (64-dim) on COCO val, colored by class (top-10 most frequent). Embeddings show meaningful cluster structure despite aggressive dimensionality reduction.

D.2 EMBEDDING VISUALIZATION

Figure 10 visualizes the learned embeddings using t-SNE.

D.3 DATASET COMPARISON

Figure 11 plots the 256d row of Table 1: TINYVLM against the CLIP teacher and chance on all six evaluation datasets.

D.4 QUALITATIVE PREDICTIONS

Figure 12 shows example predictions from TINYVLM across five of the six evaluation datasets (all but LVIS).

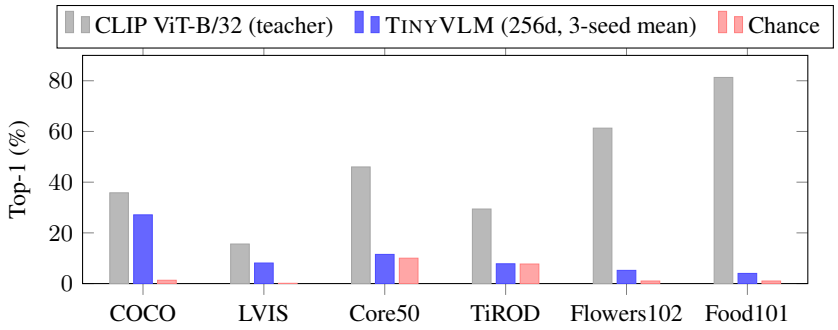

Figure 11: Zero-shot top-1 at 256d on each evaluation dataset, against the CLIP ViT-B/32 teacher and chance: the numbers of Table 1. Retention falls with granularity; corrected for chance, Core50 (4%) sits with the fine-grained sets and TiROD carries no signal (Section 4.5).

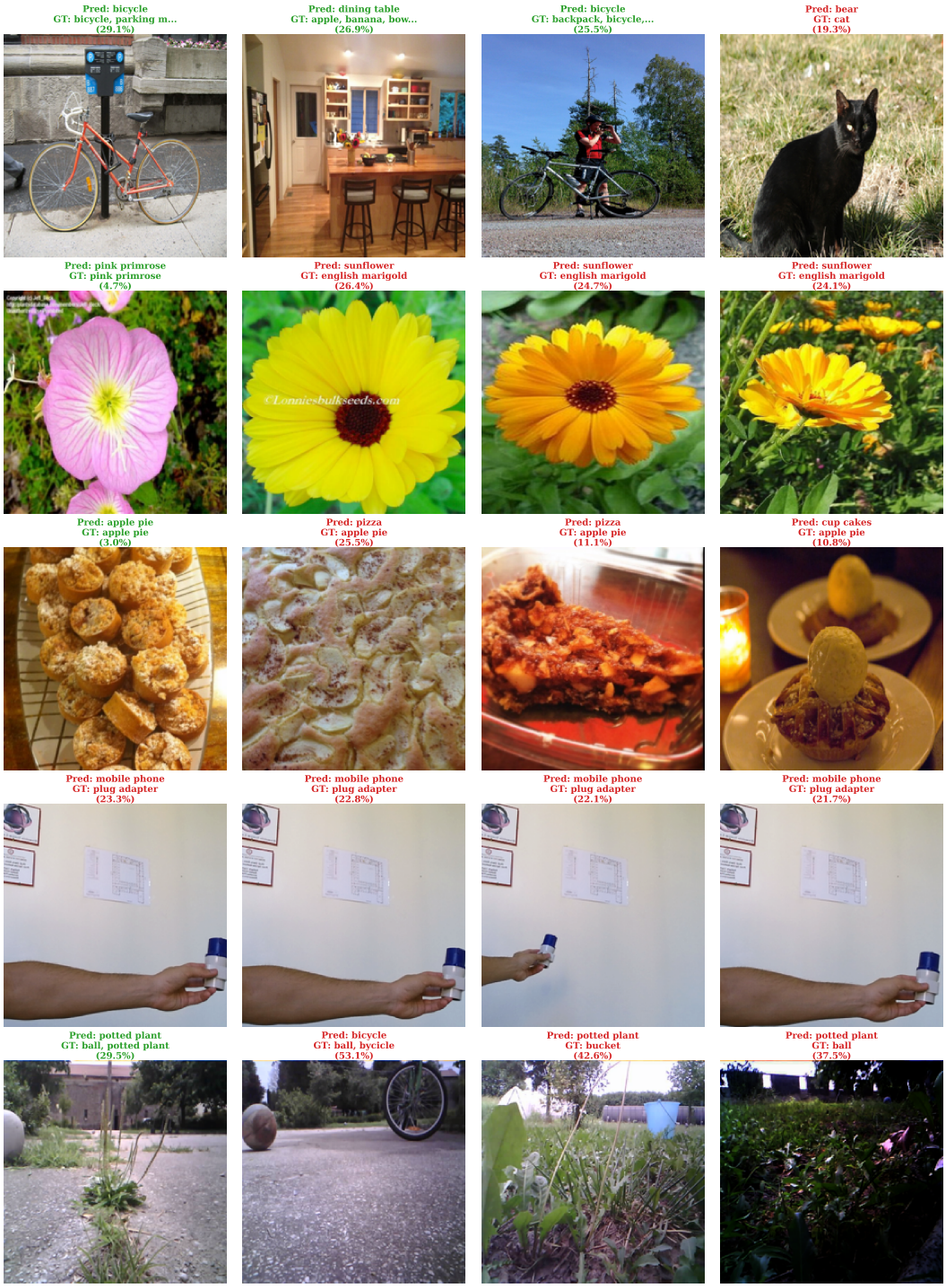

Figure 12: Qualitative zero-shot predictions from TINYVLM (256-dim). For each dataset, we show representative correct (green) and incorrect (red) predictions sorted by confidence. General object recognition succeeds on COCO while fine-grained datasets remain challenging.