LEARN AND PIN: A TRANSFERABLE JEPA-STYLE VISUAL TOKEN COMPRESSOR FOR FROZEN VLM HOSTS

Anonymous authors
Paper under double-blind review

ABSTRACT

In vision–language models, the image dominates the input: LLaVA-1.5 renders one image as 576 tokens, far more than a typical prompt, so most image compute is spent inside the language model. Existing visual token compressors either hand-specify the pre-designed method or learn it with task supervision tied to one host. None learns label-free model inside a frozen encoder and transfers it, unmodified, to a new host. In this paper, we train a small compressor inside a *frozen* vision encoder without captions. It uses a *JEPA-style* predictive term and a *distributional pin*. The pin is a kernel two-sample statistic that keeps the distribution of the compressed tokens close to the encoder's own. The compressor then runs frozen in the encoder's forward pass and is evaluated zero-shot. We do not quote published numbers. We re-run all competitors, both published methods and training-free anchors, in one setting. They are matched on token count, or on total cost for decoder-side pruners. At $K{=}144$ (a quarter of the tokens) on the LLaVA + SmolLM2-135M parameter host, the proposed method leads the field by $+9.02 \pm 1.23$ CIDEr zero-shot and $+11.08 \pm 2.76$ one-shot at 43.3% of the uncompressed analytic cost. The compressor transfers unchanged into LLaVA-1.5-7B. It retains 92–97% of the uncompressed ceiling on five of six in-context tasks and 94–99% on five standard benchmarks, at 31% of the analytic cost. It also transfers to an unseen Q-Former bridge at 96.6% retention, $+5.02$ over the strongest competitor. On CLIP retrieval, it retains 99.5% of recall at $K{=}98$ and leads TOME by $+0.72$.

1 INTRODUCTION

In vision–language models (VLMs), the image takes up most of the language model's context window. LLaVA-1.5 (Liu et al., 2024a), for example, passes 576 patch tokens from a frozen CLIP ViT-L/14-336 vision encoder (Radford et al., 2021) to a 7B decoder. Processing these tokens costs about ten times more than running the vision encoder that produced them.Prior work on this problem falls into two broad families. *Training-free* compressors apply pre-designed compression algorithms rather than learned ones, whereas *trained* compressors learn through task supervision. Neither group offers all three of the following properties. The compression model should be *learned* from data rather than designed by hand. It should need *no task labels*, such as captions or class annotations. It should not be tied to one host. Pre-designed methods are fragile (Wen et al., 2025; Endo et al., 2025), labels discard image detail that other tasks need (Oquab et al., 2024), and dependence on one host forces costly retraining for every new model (Zhang et al., 2023). BLIP-2's Q-Former (Li et al., 2023b) comes closest: it trains a compressor while the image encoder and the language model stay frozen. But it is fit with image–text supervision against one host, so it is neither label-free nor portable to a host it was not fit against. A compressor trained without labels and then run frozen inside a host it was never fit to remains an open problem.

In this paper, we train a compressor inside a *frozen* vision encoder without any labels. We then use the compressor as-is, with no further adaptation, inside the host model's forward pass. Our objective has two terms (Section 3). (1) A JEPA-style predictive term trains the compressed tokens to carry enough information to predict masked feature-layer content. (2) A *distributional pin*, a kernel two-sample statistic, keeps the distribution of the compressed tokens close to that of the frozen tower.

The contributions of this paper can be summarized as:

- **A JEPA-style visual token compressor** that trains once on unlabeled data and drops into any frozen vision encoder with no retraining required, combining the efficiency of a one-time

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.

training cost with complete independence from labeled supervision. To our knowledge, no prior visual token compressor uses this combination.

- **A comprehensive evaluation study** built on matched-geometry comparisons where all baselines and relevant state-of-the-art methods are re-implemented and evaluated under identical configurations and token budgets.

The rest of this paper is organized as follows: Section 2 reviews prior work. Section 3 describes the method. Section 4 describes the experimental setup. Section 5 reports the main results. Section 6 reports the ablation studies. Section 7 discusses when the method helps and when it fails. Appendix A lists the limitations.

2 Related Work

As mentioned above, the prior work on visual token compression falls into two groups: training-free and trained methods. We review both groups below and the JEPA models that inspired our method.

Training-free methods follow two main approaches: token merging and token pruning. Token merging approaches combine similar tokens into one, usually by averaging them. For instance ToMe (Bolya et al., 2023) merges similar tokens in each block of the vision encoder using bipartite matching. In contrast, token pruning scores each token by an importance measure, usually attention, and drops the tokens with the lowest scores. For instance, FastV (Chen et al., 2024) and SparseVLM (Zhang et al., 2025) drop the image tokens that receive the least attention score. Some methods combine both approaches like VisionZip (Yang et al., 2025) and PruMerge+ (Shang et al., 2025), which first keep a small set of key tokens, chosen by the attention of the class token, then merge the remaining tokens instead of dropping them. VisionZip merges them into a few new context tokens, while PruMerge+ merges them into the kept tokens.

Trained compressors learn the compression task from data. Most of them use task labels, such as captions or instructions, and are trained for one host model. For example, CrossGET (Shi et al., 2024) uses task supervision and also fine-tunes the host. Other methods compress the tokens at the bridge, the module that connects the vision encoder to the language model (Hu et al., 2024; Li et al., 2025; 2023b), or inside the language decoder (Ye et al., 2025).

JEPA objectives predict masked content in latent space (Assran et al., 2023) and have been used to build compact speech and image tokens (Ioannides et al., 2025; Baek et al., 2025). Predicting latent vectors has also been used to train visual token compressors, often with a fine-tuned language model (Hou et al., 2026; Jin et al., 2024; Liu et al., 2025; Gao et al., 2026). LeJEPA (Balestriero & LeCun, 2025) regularizes the embeddings by pulling them toward a isotropic gaussian distribution, whereas KerJEPA (Zimmermann et al., 2025) shows that isotropic gaussian regularization has an impact similar to sliced Maximum Mean Discrepancy (MMD)-based regularizer. In our proposed work, we use MMD to match the frozen encoder's token distribution to the summarized tokens. Our compressor also runs unchanged in frozen hosts it never saw in training.

3 Method

Our proposed method consists of a single module spliced into a frozen vision encoder and trained against that encoder's own activations. Nothing downstream is adapted to accommodate it, and no term in its objective function depends on a label.

3.1 Inserting the Compressor into the Frozen Encoder

Let $\mathcal{V}$ be a frozen vision transformer (Dosovitskiy et al., 2021) of depth L width C, and N patch tokens. Let τ denote the host feature layer, the index of the block whose output is consumed by the next component. For CLIP, $\tau = L = 12$, and for the generative hosts, $\tau = 23$.

A compressor $\Sigma : \mathbb{R}^{B\times N\times C} \to \mathbb{R}^{B\times K\times C}$ is an information bottleneck with $K \ll N$, where B is the batch size, N and K are the input and output token counts, and C is the channel width. Let n be the number of tokens that enter the compressor ($n = N$ at the first splice). It acts on patch tokens only, and the class token is routed around it unchanged. It is inserted before a selected block $\ell_j \leq \tau$, so that this block already runs at the compressed length:

$$\mathbf{X}^{(\ell_j)} \longmapsto \left[\mathbf{X}^{(\ell_j)}_{:,0:1,:} \parallel \Sigma_j\big(\mathbf{X}^{(\ell_j)}_{:,1:,:}\big)\right]. \tag{1}$$

3.2 COMPRESSOR ARCHITECTURES

We define two $N \to K$ compression learned architectures; attention-pool and grid-pool. All experiments executed in this paper use one of the two architectures or both.

Attention-pool We follow the work presented in (Ba et al., 2016; Xiong et al., 2020) in using pre-LN blocks in pattern learning. Given X input uncompressed tokens, we stack $L_{compressor}$ pre-LN blocks, $\mathrm{Blocks}_\theta(X)$, that keep the number of output tokens $\widetilde{\mathbf{X}}$ unchanged,

$$\widetilde{\mathbf{X}} \;=\; \mathrm{Blocks}_\theta(\mathbf{X}) \;\in\; \mathbb{R}^{B\times n\times C}, \tag{2}$$

followed by a cross-attention layer with K learned query vectors $\Theta \in \mathbb{R}^{K\times C}$. We define the *content branch* $\mathbf{U}$ to be used in deriving the summarization tokens:

$$\mathbf{U} \;=\; W_o\,\mathrm{MHA}\big(W_q\mathrm{LN}(\Theta),\; W_k\mathrm{LN}(\widetilde{\mathbf{X}}), W_v\mathrm{LN}(\widetilde{\mathbf{X}})\big) \;\in\; \mathbb{R}^{B\times K\times C}, \tag{3}$$

where MHA is standard H-head attention and $W_q, W_k, W_v, W_o \in \mathbb{R}^{C\times C}$ are the projection matrices. We apply Rotary embeddings (Su et al., 2024) only to the keys, since the queries have no fixed position. There are K queries, so this module supports any output size. We define anchor $\mathbf{A}$ as top-k tokens selected by CLS attention scores of the consuming block. The summarized tokens $\mathbf{Z}$ are generated by combining the anchor $\mathbf{A}$ with two zero-gated branches,

$$\mathbf{Z} = \mathbf{A} + \gamma_{\mathrm{a}} \odot \mathbf{U} + \gamma_{\mathrm{m}} \odot \mathrm{MLP}\big(\mathrm{LN}(\mathbf{A} + \gamma_{\mathrm{a}} \odot \mathbf{U})\big), \tag{4}$$

where $\gamma_{\mathrm{a}}, \gamma_{\mathrm{m}} \in \mathbb{R}^C$ are per-channel LayerScale gates, as proposed by Touvron et al. (2021), initialized to zero. Each compressed token is thus a real input token plus a learned offset.

Grid-pool Unlike the Attention-pool architecture, we reduce the number of tokens with two-dimensional spatial pooling and then refine the result with learnable layers. Specifically, given n uncompressed input tokens $\mathbf{X}$, we reshape the n tokens into a dense $g\times g$ grid ($n = g^2$) in row-major order, so token i occupies position (r, c) with $i = rg + c$, where grid expresses this re-indexing function and vec expresses its inverse. We then apply a single 2×2 average-pooling $\mathcal{P}$ step with stride 2. We refine the pooled tokens with $L_{compressor} \geq 1$ pre-LN blocks following the form in Eq. (2) similar to Attention-pool architecture. The summarized tokens $\mathbf{Z}$ are

$$\mathbf{Z} \;=\; \mathrm{Blocks}_\theta\big(\mathrm{vec} \circ \mathcal{P} \circ \mathrm{grid}(\mathbf{X})\big) \;\in\; \mathbb{R}^{B\times K\times C}. \tag{5}$$

In contrast to the attention-pool architecture, grid pool fixes K to $n/4$. Each compressed token is thus the mean of a 2×2 window of input tokens plus a learned refinement.

3.3 TRAINING OBJECTIVE

The above-mentioned architectures are trained with an objective function that consists of two terms: a JEPA-style predictive term and a distributional pin. To define these terms, let $\mathbf{T} \in \mathbb{R}^{B\times n\times C}$ be the encoder's patch tokens at the same feature layer, with $\mathbf{T}$ taken under a stop-gradient $\mathrm{sg}[\cdot]$ (Chen & He, 2021).

Predictive term This term follows JEPA-style decoding: masked positions are decoded from the compressed tokens alone (Assran et al., 2023; Bardes et al., 2024). At each step, we sample a hidden set $\mathcal{M}$ of token positions uniformly at random. This set is shared across the batch, and its size is fixed at a fraction 0.5 of the n positions. Each masked position i produces a query $\mathbf{q}_i = \mathbf{m} + \mathrm{pe}(i)$, where $\mathbf{m}$ is a learned mask token shared by all positions, and $\mathrm{pe}(i)$ is the fixed 1-D sinusoidal positional encoding (Vaswani et al., 2017) of the true index i. We denote this term $\mathcal{L}_{\mathrm{pred}}$, which is defined as:

$$\mathcal{L}_{\mathrm{pred}} = \ell_{\mathrm{SL1}}\Big(W_p\big[\mathrm{Predictor}_\phi([\mathbf{Z} + \mathbf{s}\,;\, \mathbf{Q}_{\mathcal{M}}])\big]_{:,K:,:}, \mathrm{LN}\big(\mathrm{sg}[\mathbf{T}_{:,\mathcal{M},:}]\big)\Big), \tag{6}$$

where, $\mathbf{Q}_{\mathcal{M}}$ is the stack of all queries q_i, and the $Predictor$ consists of D_{pred} pre-LN blocks followed by a linear head W_p. The term $\mathbf{s}$ is a zero-initialized segment embedding that marks the compressed rows. LN normalizes each target token, and ℓ_{SL1} denotes the smooth-L_1 loss. The targets $\mathbf{T}$ come from the frozen encoder and cannot collapse as it is not learnable, so no anti-collapse mechanism is needed.

Distributional pin The predictive term alone does not keep the compressed tokens on the frozen encoder's token manifold, which the host was trained on. We, therefore, add a second term, the unbiased RBF-kernel MMD^2 (Gretton et al., 2012), that enforces this alignment. Specifically, we first reduce the compressed and uncompressed tokens, $\mathbf{A_S}$ and $\mathbf{A_T}$ respectively, to at most $M_{\mathrm{cap}} = 1{,}024$

rows by a *deterministic* strided subsample. This gives $\mathbf{u}_1, \ldots, \mathbf{u}_m$ from $\mathbf{A_S}$ and $\mathbf{v}_1, \ldots, \mathbf{v}_{m'}$ from $\mathrm{sg}[\mathbf{A_T}]$, with $m, m' \le M_{\mathrm{cap}}$. Let $k_\sigma(\mathbf{x}, \mathbf{y}) = \exp\big(-\|\mathbf{x}-\mathbf{y}\|_2^2/2\sigma^2\big)$ denote the RBF kernel. The estimator at a single bandwidth $\widehat{\mathrm{MMD}}^2_\sigma$ and the distributional pin term $\mathcal{L}_{\mathrm{pin}}$ can be defined as:

$$\widehat{\mathrm{MMD}}^2_\sigma = \frac{1}{m(m-1)} \sum_{i \neq i'} k_\sigma(\mathbf{u}_i, \mathbf{u}_{i'}) \;+\; \frac{1}{m'(m'-1)} \sum_{j \neq j'} k_\sigma(\mathbf{v}_j, \mathbf{v}_{j'}) \;-\; \frac{2}{m\,m'} \sum_{i,j} k_\sigma(\mathbf{u}_i, \mathbf{v}_j),$$

$$\mathcal{L}_{\mathrm{pin}} = \frac{1}{|\mathcal{B}|} \sum_{\sigma \in \mathcal{B}} \widehat{\mathrm{MMD}}^2_\sigma, \qquad \mathcal{B} = \{0.5,\, 1,\, 2\} \cdot \sigma_{\mathrm{med}}. \tag{7}$$

We drop the self-kernel diagonals so that each term is unbiased. The scale σ_{med} is the median of the nonzero pairwise distances among the capped rows of $\mathrm{sg}[\mathbf{A_T}]$ only, and it is re-estimated on every batch. As a result, the kernel follows the scale of each batch's target tokens, and the total loss can be finally expressed as:

$$\mathcal{L} \;=\; w_{\mathrm{pred}}\, \mathcal{L}_{\mathrm{pred}} \;+\; \lambda\, \mathcal{L}_{\mathrm{pin}}. \tag{8}$$

4 Experimental Setup

This section describes the frozen host models, the datasets, the baselines, the implementation details and the evaluation protocols used in all experiments.

4.1 Host Models

We evaluate the proposed method on three frozen hosts: CLIP, LLaVA, and a BLIP-2-variant. The first host is **CLIP** ViT-B/16 (N=196, τ=12) which is a dual encoder for joint image–text retrieval (Radford et al., 2021). **LLaVA**, the second host, pairs a frozen CLIP ViT-L/14-336 encoder (N=576, τ=23) with an MLP projector as in the LLaVA-1.5 configuration (Liu et al., 2024a). Two decoder scales are used for LLaVA: SmolLM2 135M (Ben Allal et al., 2025) as the *research* host, which we use to train the proposed compressor and transfer it to other hosts. We use the released LLaVA-1.5-7B bundled with Vicuna-7B (Chiang et al., 2023) and its own trained projector (Liu et al., 2024a) as the *deployment* host, which is used for evaluating the transferability of the proposed compressor. To test whether the compressor depends on the bridge, we add a third host, a **BLIP-2** variant. It matches the LLaVA research host in every component except the bridge. The MLP projector is replaced by a Q-Former (Li et al., 2023b) initialized from BERT-base (Devlin et al., 2019), with no stage-1 pretraining. The BLIP-2-variant host is our own version, not the released Salesforce model (Li et al., 2023b;a). The CLIP host is used for an **image–text retrieval** experiment track while the LLaVA and BLIP-2 are used for **captioning tasks**.

4.2 Datasets

Pretraining data. The compressor is pretrained on images only with no caption needed. The image–text retrieval experiments use the Flickr30k training split of that has 29k images (Karpathy & Fei-Fei, 2015). The captioning experiments use the COCO Karpathy training split with 113k images (Lin et al., 2014).

Evaluation data. We evaluate our work on four task types. *1)* Retrieval tasks use Flickr30k. *2)* Captioning tasks use the COCO Karpathy split and is scored with CIDEr (Vedantam et al., 2015) throughout. *3)* In-context question answering tasks use VQAv2 (Goyal et al., 2017) and OK-VQA (Marino et al., 2019). *4)* Episodic few-shot classification tasks use two fine-grained datasets: CUB-200-2011 (Wah et al., 2011) and FGVC-Aircraft (Maji et al., 2013). On the deployment host, we also report six standard LLaVA-1.5 benchmarks: POPE (Li et al., 2023c), GQA (Hudson & Manning, 2019), MME (Fu et al., 2025), MMBench (Liu et al., 2024b), SEED (Li et al., 2024) and TextVQA (Singh et al., 2019) (Section 5.2).

4.3 Competing Techniques and Baselines

We compare our work against five published training-free methods: ToMe, VisionZip, PruMerge+, FastV and SparseVLM. We re-implement all five and run them with the settings of their original papers. ToMe also runs in the one-shot and cost-matched variants listed in Table 2. FastV and SparseVLM prune tokens inside the decoder, we therefore match them on total cost rather than on token count. We also run six training-free baselines (anchors): CLS top-k, strided selection, random selection at three seeds, and the 2×2 average pooling. Furthermore, for the

Table 1: Evaluation protocols.

Tag	Evaluation	Split	Size
C-MON	COCO captioning, greedy	Karpathy val subset	1,000 images
ICL	in-context captioning, COCO	Karpathy test	968 episodes
	in-context VQA, VQAv2 / OK-VQA	COCO val2014	1,967 / 1,968 q.
E-600	episodic 5-way, 1-shot, 5-shot	CUB, Aircraft test	600 episodes
B-7B	LLaVA-1.5 suite, six benchmarks	capped subsamples, MME whole	1,805–2,802 q.
R-VAL	Flickr30k retrieval, $\bar{r}$ (monitor)	Karpathy val	1,014 images
R-TEST	Flickr30k retrieval, $\bar{r}$	seeded 10% partition	3,178 images

Table 2: Main results. Each row compares our compressor with the strongest competitor in that cell, selected post hoc. On the deployment host, only token-matched modes compete, since FASTV and SPARSEVLM are matched on total cost. The BLIP-2 $K{=}288$ row is a bridge-isolation control, not an efficiency result.

Task	Geometry	Prot.	Ceiling	Ours	Ret.%	Best competitor	Score	Δ
Research host, LLaVA + SmolLM2-135M								
COCO CIDEr, 0-shot	grid $K{=}144$	ICL	117.84	**100.10**	84.9	random ($s{=}0$)	91.08	+9.02
COCO CIDEr, 1-shot	grid $K{=}144$	ICL	98.91	**60.50**	61.2	random ($s{=}0$)	49.42	+11.08
COCO CIDEr, 0-shot†	pool $K{=}288$	ICL	117.84	**112.88**	95.8	random ($s{=}2$)	108.54	+4.34
COCO CIDEr, 1-shot†	pool $K{=}288$	ICL	98.91	**72.97**	73.8	TOME (one-shot)	72.58	+0.39
Deployment host, LLaVA-1.5-7B, no host adaptation								
VQAv2 acc, 0-shot	grid $K{=}144$	ICL	76.48	71.71	93.8	TOME	72.58	−0.87
VQAv2 acc, 1-shot	grid $K{=}144$	ICL	74.18	**70.20**	94.6	TOME	70.16	+0.04
OK-VQA acc, 0-shot	grid $K{=}144$	ICL	58.94	**56.95**	96.6	random ($s{=}1$)	55.40	**+1.55**
OK-VQA acc, 1-shot	grid $K{=}144$	ICL	56.23	**54.05**	96.1	random ($s{=}1$)	52.21	**+1.85**
COCO CIDEr, 0-shot	grid $K{=}144$	ICL	58.16	**53.51**	92.0	TOME	53.03	+0.48
COCO CIDEr, 1-shot	grid $K{=}144$	ICL	44.86	**37.42**	83.4	TOME	37.36	+0.06
VQAv2 acc, 0-shot†	pool $K{=}288$	ICL	76.48	72.50	94.8	CLS top-k	75.83	−3.33
OK-VQA acc, 0-shot†	pool $K{=}288$	ICL	58.94	57.50	97.6	VISIONZIP	58.45	−0.95
COCO CIDEr, 0-shot†	pool $K{=}288$	ICL	58.16	55.72	95.8	random ($s{=}0$)	56.27	−0.55
Q-Former bridge, BLIP-2 host								
COCO CIDEr, 0-shot†	grid $K{=}144$	ICL	99.90	**96.51**	96.6	TOME	91.49	+5.02
COCO CIDEr, 0-shot†	pool $K{=}288$	ICL	99.90	**100.45**	100.6	VISIONZIP	98.98	+1.47
Dual encoder, CLIP ViT-B/16, $N{=}196$								
Flickr30k $\bar{r}$†	pool $K{=}98$	R-TEST	77.55	**77.19**	99.5	TOME	76.47	+0.72
Flickr30k $\bar{r}$†	grid $K{=}49$	R-TEST	77.55	**72.95**	94.1	TOME	72.39	+0.56

ablation studies, we use the untrained compressor at the same geometry as a further control, which we call the untrained twin. We do not quote published numbers, rather every method runs in one evaluator with the same code and data, following (Musgrave et al., 2020) and (Haurum et al., 2023).

4.4 Optimization and Evaluation Protocols

Only the compressor is optimized. We use AdamW (Loshchilov & Hutter, 2019) with learning rate 5×10^{-5}, weight decay 0.05, and a 200-update linear warmup and cosine decay (Loshchilov & Hutter, 2017). We clip gradients at 1.0 and train in bf16 mixed precision (Micikevicius et al., 2018). We hold out 5% of the training images for hyperparameter optimization. The evaluations use several protocols, as illustrated in Table 1. These protocols differ in dataset size, split ratio, and decoding harness, so each protocol has its own uncompressed ceiling representing which is defined as the score of the same frozen host, run through the same evaluator and protocol, with no compressor at all. Each results table below tags the protocol used referenced in Table 1. Scores with different tags are not comparable.

5 Experimental Results

This section compares the trained compressor with the competing techniques and baselines. In each comparison, the geometry, budget, host and evaluator are fixed, so only the compression method differs. We train the compressor once on the research host. We then insert it, unchanged, into LLaVA-1.5-7B and the BLIP-2 host. The CLIP host, on the other hand, uses a different vision encoder, so we train its compressor on CLIP itself.

Table 3: The standard LLaVA-1.5 benchmarks, zero-shot on LLaVA-1.5-7B at the grid $K{=}144$ geometry.

Mode	Class	POPE acc	POPE F1	GQA	MMBench	MME acc	SEED	TextVQA
uncompressed ceiling	ceiling	86.37	84.81	62.44	69.51	78.83	60.79	49.30
CLS top-k	anchor	84.48	82.63	56.34	65.65	73.59	54.77	37.10
random ($s{=}0$)	anchor	82.12	79.10	58.78	65.35	74.81	55.63	21.36
random ($s{=}1$)	anchor	82.48	79.38	57.56	66.06	74.20	55.58	21.10
$2{\times}2$ avg-pool	anchor (free)	80.26	75.97	56.29	64.13	74.16	53.24	17.10
ToMe	published	86.26	85.14	59.56	65.85	76.41	55.02	33.33
VisionZip	published	84.33	82.52	57.12	65.80	73.81	55.07	35.44
PruMerge+	published	81.16	77.46	55.12	64.13	72.94	52.32	22.10
FastV	published	77.77	71.28	53.74	68.14	76.58	54.46	45.62
SparseVLM	published	77.98	71.63	53.24	67.68	76.84	54.21	45.48
ours (pred + pin)	ours	**85.19**	**83.33**	**58.62**	**65.40**	**75.50**	**57.47**	**22.16**
best token-matched is		ToMe	ToMe	ToMe	random ($s{=}1$)	ToMe	random ($s{=}0$)	CLS top-k
Δ vs. best of any kind		-1.07	-1.81	-0.94	-2.74	-1.34	$+1.84$	-23.45
Δ vs. best token-matched		-1.07	-1.81	-0.94	-0.66	-0.91	$+1.84$	-14.94
Δ_{learn} vs. avg-pool		$+4.93$	$+7.36$	$+2.33$	$+1.27$	$+1.34$	$+4.23$	$+5.06$
Retention vs. ceiling (%)		98.6	98.3	93.9	94.1	95.8	94.5	45.0

Captioning is the task that most directly tests whether compressed tokens still carry what a decoder needs, and in this respect our label-free compressor, placed inside a frozen encoder, keeps most of the uncompressed quality on every host we test. To quantify this, Table 2 reports retention against the uncompressed ceiling together with the margin over the strongest competing mode, selected post hoc per cell, at both a quarter of the tokens (grid) and half of them (pool). Overall, retention stays high in all but two cells and, with BLIP-2, even matches the ceiling. The only exceptions are one-shot captioning on the small research host. Turning to the margins, the advantage at the quarter budget is largest on the research host, and this holds even though its strongest competitor, random selection, already retains most of the zero-shot ceiling. On the larger deployment host, by contrast, the margin is smaller: the gains on OK-VQA are significant, whereas VQAv2 shows one significant loss. Moreover, at half the tokens the margin shrinks on every decoder host and turns negative on the deployment host. Taken together, these results show that the compressor preserves the semantics that captioning and question answering require across scales, budgets, and unadapted bridges, while its advantage over competing compressors is concentrated at the tighter budget.

5.1 Transfer to a Larger Host and to a Q-Former Bridge

The objective function depends only on the frozen vision encoder. A compressor trained with one host and one bridge should therefore can be transferred to a larger host or a different bridge without adaptation. To verify this claim, we test with the same compressor developed on the 135M parameters host and transferred unchanged. The second section in Table 2 reports six in-context tasks on LLaVA-1.5-7B with retention 92.0–96.6% of the ceiling on five tasks and 83.4% on one-shot captioning. The margins over the strongest competitor are small, from -0.87 to $+1.85$. Our method trails ToMe on zero-shot VQAv2 and leads on the other five tasks. Table 2 (BLIP-2 part) and Table 5 report the same trained compressor inside the Q-Former host. The zero-shot cell retains 96.6% of the ceiling and leads every competitor.

Two things to observe here. First, the transfer is meaningful only because both hosts share the same vision encoder. Second, since the Q-Former emits a fixed number of queries regardless of K, the $K{=}288$ experiment probes the effect of K at the bridge itself, rather than isolating the performance gain attributable to K tokens. Thus, the proposed work produces a compressor that can be transfer to a large model regardless of the bridge type, and without retraining nor significant performance degradation.

5.2 Standard LLaVA-1.5 Benchmarks

The standard suite of LLaVA benchmark is where retention could change. However, high retention of the ceiling does not imply a lead because the performance measure scored by CIDEr is close among all competing techniques on most of these benchmarks. Table 3 reports the main geometry on POPE, GQA, MME, MMBench, SEED and TextVQA. It compares our proposed work against the token-matched encoder-side competitors, the parameter-free pool, and the decoder-side pruners, which are matched on total cost.

On the first five benchmarks, the trained compressor sits within the recorded performance range. However, it lags behind the strongest token-matched competitor narrowly on four benchmarks, leads

Table 4: Flickr30k retrieval on the seeded R-TEST partition, two CLIP geometries, every mode.

Mode	Class	pool $K{=}98$	grid $K{=}49$
uncompressed ceiling	ceiling	77.55	77.55
CLS top-k	anchor	75.79	37.03
strided	anchor	73.57	57.87
random ($s{=}0$)	anchor	73.37	58.11
random ($s{=}1$)	anchor	72.30	55.98
random ($s{=}2$)	anchor	73.09	52.96
$2{\times}2$ avg-pool	anchor (free)	—	46.13
untrained twin	control	75.79	0.43
ToMe	published	76.47	72.39
ToMe (one-shot)	published	74.81	57.24
VisionZip	published	75.72	49.44
PruMerge+	published	1.55	54.54
ToMe (cost-matched)	published	74.13	66.71
ours (pred + pin)	ours	**77.19**	**72.95**
Δ vs. best competitor		$+0.72$	$+0.56$
Retention vs. ceiling (%)		99.5	94.1

Table 5: Gain from label-free training beyond the pooling alone, at the same geometry, splice point and K, and nearly the same cost. Our scores come from a single evaluation run.

Host, arm	Task	Prot.	free	ours	Δ_{learn}
Grid family — the free compressor is the $2{\times}2$ average pool, which carries no parameters					
LLaVA (res.), grid $K{=}144$	COCO CIDEr, 0-shot	ICL	78.39	99.05	$+20.66$
	COCO CIDEr, 1-shot	ICL	39.81	59.94	$+20.13$
	CUB acc, 1-shot	E-600	51.24	56.86	$+5.62$
	CUB acc, 5-shot	E-600	71.35	78.01	$+6.67$
	Aircraft acc, 1-shot	E-600	40.85	47.91	$+7.06$
	Aircraft acc, 5-shot	E-600	53.87	62.17	$+8.30$
Deployment host — same free compressor, same trained weights, zero-shot transfer					
LLaVA-1.5-7B, grid $K{=}144$	VQAv2 acc, 0-shot	ICL	65.53	71.61	$+6.08$
	OK-VQA acc, 0-shot	ICL	52.91	57.04	$+4.14$
	COCO CIDEr, 0-shot	ICL	45.74	54.10	$+8.36$
	POPE acc	B-7B	80.26	85.19	$+4.93$
	POPE F1	B-7B	75.97	83.33	$+7.36$
	GQA	B-7B	56.29	58.62	$+2.33$
	MMBench	B-7B	64.13	65.40	$+1.27$
	MME acc	B-7B	74.16	75.50	$+1.34$
	SEED	B-7B	53.24	57.47	$+4.23$
	TextVQA	B-7B	17.10	22.16	$+5.06$
Second bridge and dual encoder — same construction, other hosts					
BLIP-2, grid $K{=}144$	COCO CIDEr, 0-shot	ICL	80.23	96.51	$+16.28$
CLIP, grid $K{=}49$	Flickr30k $\bar{r}$	R-TEST	46.13	72.95	$+26.81$

on SEED, and exceeds the parameter-free pool on all five. The decoder-side pruners lead every encoder-side method on MMBench and MME. We observe that our proposed method performs close to published results on these benchmarks, within difference of 2.7 CIDEr points at most. On TextVQA, our performance is weaker, but this degradation is consistent across all similar encoder-based methods.

5.3 Retrieval with CLIP

Retrieval pools the K tokens into one joint-space embedding. This limits the impact the compressor can do, so small margins are expected on this task. Table 4 reports two geometries on the retrieval host: a four-stage joint cascade at $K{=}98$ and a single-splice grid at $K{=}49$. Each is compared against the uncompressed ceiling, the strongest competitor, and its own parameter-free initialization. The method retains 99.5% of the ceiling at $K{=}98$ and 94.1% at $K{=}49$. It outperforms ToMe by 0.72 and 0.56 points in the two summarizers respectively. We can observe that our proposed work outperforms all published works and baselines.

6 Ablation Studies

This section isolates three factors. Section 6.1 removes training and keeps the geometry. Section 6.2 varies the token budget. Section 6.3 trains each loss term alone.

6.1 Trained Compressor versus Parameter-free Control

A trained compressor may owe part of its quality to its fixed structure rather than to training. We therefore compare the compressor trained with Eq. (8) against a parameter-free control that uses the same $2{\times}2$ average pool, token grid, border rule, splice point and K, but lacks the refinement blocks. As Table 5 shows, training improves the score in all eighteen settings across four hosts. The gains are largest on captioning and retrieval, up to $+26.81$ mean recall for CLIP, and smaller on fine-grained classification ($+5.62$ to $+8.30$). On the deployment host, where the weights are reused without retraining, the gain on COCO falls from $+20.66$ to $+8.36$ and is smallest on MMBench ($+1.27$), yet

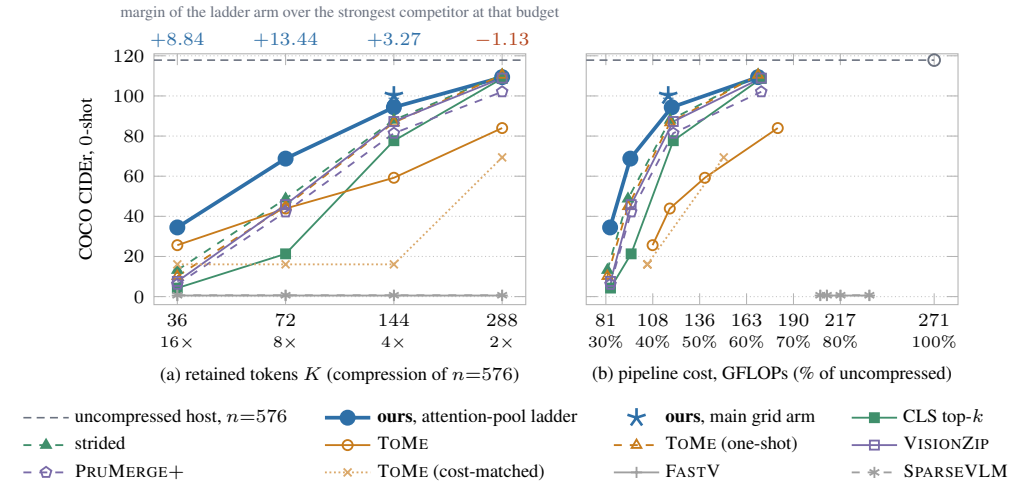

Figure 1: **The advantage is a window in the token budget.**

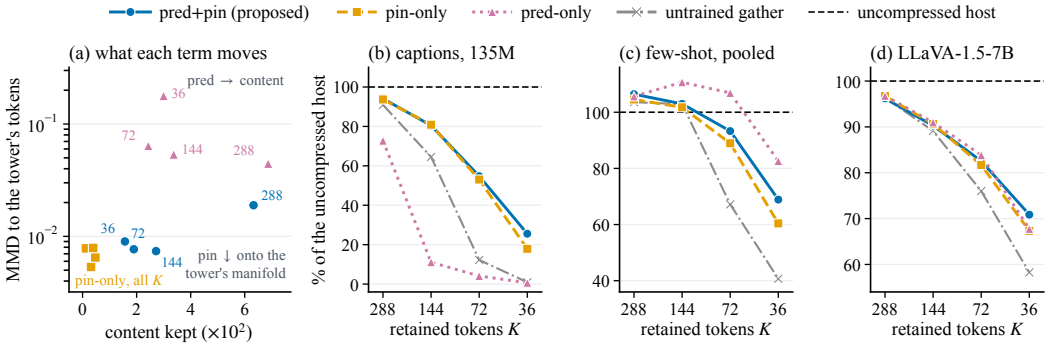

Figure 2: **The two terms move the tokens along different axes.** Attention-pool compressor, one training seed per setting. (a) Content gain over the untrained gather against held-out MMD to the encoder's tokens. (b)–(d) Scores as a percentage of the uncompressed host: (b) captions on the research host (C-MON), (c) mean CUB and Aircraft 5-shot accuracy (E-600), and (d) mean retention over nine LLaVA-1.5-7B tasks.

it stays positive on every task. Hence, the compressor's quality does not come from the pooling alone: label-free training adds a consistent gain at nearly the same cost, and part of this gain carries over to a host the compressor was not trained on.

6.2 Effect of the Token Budget

A learned compressor can only beat a training-free method where that method discards information the task needs, so its benefit should vary with the token budget. To test this, Figure 1 plots zero-shot COCO CIDEr on the research host against the number of kept tokens K (a) and the pipeline cost (b), with all competitors re-run at each budget. At half the tokens (K=288), the training-free methods are already close to the uncompressed host (117.84), leaving little for learning to recover. As the budget shrinks, however, they lose accuracy faster than the learned compressor, and the gap widens. At the smallest budgets, the learned compressor still outperforms others, but its score becomes too low to be useful. Taken together, the benefit of learning is a window $72 - 144$ of compression ratios rather than a lead at every budget.

6.3 Contribution of Each Loss Term

Compressed tokens must both carry the image content and keep the form that the frozen downstream blocks and decoder can read. The two terms of Eq. (8) target these needs separately (Section 3.3): the predictive term puts content into the K tokens, while the pin keeps them on the vision encoder's manifold. To measure each role of these two terms, we train the full objective and each term alone at four budgets (Figure 2). Panel (a) measures content, that is, how much better a fresh predictor rebuilds the 576 encoder tokens from the K compressed tokens than from the untrained CLS top-k

gather. It also measures manifold fit as the held-out MMD of Eq. (7). Panels (b) to (d) then score the same compressors on three tasks. Panel (a) confirms the split: the predictive term raises content, the pin lowers the MMD, and the full objective keeps much of the content at a low MMD. The tasks, however, reward these properties differently. Pooled few-shot recognition follows content and is penalized by the pin. Captioning with the small decoder, in contrast, follows the MMD. Although the predictive term alone gives the most content, its tokens caption below the untrained gather. Retraining only their output layer under the pin restores captions at $K \geq 72$, so the content was present but off the manifold. Finally, the 7B decoder reads all three objectives equally well until $K{=}36$. The full objective is therefore a hedge: it ties or beats the better single term on captions and keeps part of the predictive term's few-shot gain. Full results are in Appendix B.

7 Discussion

The benefit of the learned compressor depends on how the K tokens is processed by the host and on how much local detail the task needs. It helps most when the host reads the tokens as a sequence and the budget is tight but not extreme, as in captioning at $K{=}72$ to 144 (Section 6.2). In contrast, the summarized token benefits shrinks when the host pools the tokens into a single vector, as in retrieval and few-shot classification. In few-shot classification, for example, no decoder reads the tokens, so the pin costs some content without adding anything the task uses (Section 6.3). The compressor fails when the task depends on fine local detail, as in TextVQA, where it retains only 45% of the uncompressed score. This reflects a missing capability. Moreover, reading the tokens as a sequence is not enough on its own. All nine deployment-host tasks do so, yet TextVQA is the only one where methods clearly separate, and there every encoder-side method falls short of the decoder-side pruners. CLS top-k helps explain why. It keeps the tokens with the highest CLS attention at full resolution and scores 14.94 points above our method. Our grid, by contrast, averages every $2{\times}2$ window and thus blurs the fine detail that reading needs. Nevertheless, our trained compressor still beats the parameter-free pool by 5.06 points on this task. The failure therefore comes mainly from uniform pooling inside the encoder rather than from the training.

8 Conclusion

We presented a visual token compressor that is trained inside a frozen vision encoder without labels and then used, unchanged, inside frozen vision–language hosts. Its objective has two terms with distinct roles. The JEPA-style predictive term puts image content into the K compressed tokens, while the distributional pin keeps these tokens on the encoder's manifold so that a small decoder can read them. Our ablations show that tasks reward these roles differently: pooled tasks such as few-shot classification benefit most from the predictive term, captioning with a small decoder depends on the pin, and the full objective balances the two. Since neither term uses labels or a decoder, the compressor is not tied to any host. Against competitors re-run in the same evaluator, it leads by $+9.02$ zero-shot CIDEr at a quarter of the tokens on the research host, at 43.3% of the uncompressed analytic cost. Moreover, although it was trained on a 135M-parameter host, it transfers without adaptation to LLaVA-1.5-7B, where it retains 92–97% of the uncompressed score on five of six in-context tasks at 31% of the cost. It also transfers to an unseen Q-Former bridge, where it retains 96.6%. In short, it learns like a trained module yet transfers like a training-free one.

Ethics statement

This work relies exclusively on publicly available datasets and pretrained models, each used under its own license, and none of these assets is redistributed. The study involved no human subjects, no new annotation and no collection of personal data. The training compute consumed by each experimental arm is logged. The one foreseeable risk specific to this work concerns object hallucination: every compressor we evaluated, including our own, reduces POPE accuracy relative to the uncompressed host, and all but ToMe reduce F1. Table 3 therefore reports both metrics, since accuracy alone hides part of this loss. This loss should be weighed before a compressed host is deployed where hallucination carries a cost.

AI use statement

In accordance with the ICLR 2027 policy on the use of generative AI, we disclose that an LLM assistant was used for two purposes that fall under the policy's recommended-disclosure category: copy-editing and restructuring the manuscript for readability, and writing and editing portions of the evaluation harness, the analysis scripts and the figure-generation code. Generative AI was not used

for any activity in the required-disclosure category. The research question, the experimental design, the method and its objective, the choice of baselines and evaluation protocols, the interpretation of the results and all claims drawn from them are the authors' own, and no dataset was cleaned, filtered or reformatted by an LLM. The authors accept full responsibility for the content of this paper, including all text, claims and artifacts produced with the assistance of generative AI.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Junyeob Baek, Hosung Lee, Christopher Hoang, Mengye Ren, and Sungjin Ahn. Discrete JEPA: Learning discrete token representations without reconstruction. In *ICML 2025 Tokenization Workshop (TokShop)*, 2025.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Agustín Piqueres Lajarín, Hynek Kydlíček, Vaibhav Srivastav, et al. SmolLM2: When smol goes big – data-centric training of a fully open small language model. In *Conference on Language Modeling (COLM)*, 2025. arXiv:2502.02737.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations (ICLR)*, 2023.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision (ECCV)*, 2024.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. `https://lmsys.org/blog/2023-03-30-vicuna/`, 2023.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Mark Endo, Xiaohan Wang, and Serena Yeung-Levy. Feather the throttle: Revisiting visual token pruning for vision-language model acceleration. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A comprehensive evaluation benchmark for multimodal large language models. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2025.

Yiling Gao, Hongchen Wei, and Zhenzhong Chen. ETC: Extreme token compression via task-aware visual information distillation in VLMs. *arXiv preprint arXiv:2606.00543*, 2026.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

Joakim Bruslund Haurum, Sergio Escalera, Graham W. Taylor, and Thomas B. Moeslund. Which tokens to use? investigating token reduction in vision transformers. In *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2023.

Haowen Hou, Zhen Huang, Zheming Liang, Qingyi Si, Chenglin Li, Shuai Dong, Kele Shao, Ruilin Li, Dianyi Wang, Nan Duan, and Jiaqi Wang. AdaCodec: A predictive visual code for video MLLMs. *arXiv preprint arXiv:2606.02569*, 2026.

Wenbo Hu, Zi-Yi Dou, Liunian Harold Li, Amita Kamath, Nanyun Peng, and Kai-Wei Chang. Matryoshka query transformer for large vision-language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Georgios Ioannides, Christos Constantinou, Aman Chadha, Aaron Elkins, Linsey Pang, Ravid Shwartz-Ziv, and Yann LeCun. JEPA as a neural tokenizer: Learning robust speech representations with density adaptive attention. In *NeurIPS 2025 Workshop on Unifying Representations in Neural Models (UniReps)*, 2025.

Yang Jin, Kun Xu, Kun Xu, Liwei Chen, Chao Liao, Jianchao Tan, Quzhe Huang, Bin Chen, Chenyi Lei, An Liu, Chengru Song, Xiaoqiang Lei, Di Zhang, Wenwu Ou, Kun Gai, and Yadong Mu. Unified language-vision pretraining in LLM with dynamic discrete visual tokenization. In *International Conference on Learning Representations (ICLR)*, 2024.

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Bohao Li, Yuying Ge, Yixiao Ge, Guangzhi Wang, Rui Wang, Ruimao Zhang, and Ying Shan. SEED-Bench: Benchmarking multimodal large language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C.H. Hoi. LAVIS: A one-stop library for language-vision intelligence. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 31–41, Toronto, Canada, 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-demo.3.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning (ICML)*, 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)*, 2023b.

Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jie Qin, Jianke Zhu, and Lei Zhang. TokenPacker: Efficient visual projector for multimodal LLM. *International Journal of Computer Vision*, 133(10):6794–6812, 2025. doi: 10.1007/s11263-025-02491-7.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023c.

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)*, 2014.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a.

Xiangrui Liu, Yan Shu, Zheng Liu, Ao Li, Yang Tian, and Bo Zhao. Video-XL-Pro: Reconstructive token compression for extremely long video understanding. *arXiv preprint arXiv:2503.18478*, 2025.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. MMBench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision (ECCV)*, 2024b.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. OK-VQA: A visual question answering benchmark requiring external knowledge. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. In *International Conference on Learning Representations (ICLR)*, 2018.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision (ECCV)*, 2020.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 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)*, 2021.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22857–22867, 2025.

Dachuan Shi, Chaofan Tao, Anyi Rao, Zhendong Yang, Chun Yuan, and Jiaqi Wang. CrossGET: Cross-guided ensemble of tokens for accelerating vision-language transformers. In *International Conference on Machine Learning (ICML)*, 2024.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based image description evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Zichen Wen, Yifeng Gao, Weijia Li, Conghui He, and Linfeng Zhang. Token pruning in multimodal large language models: Are we solving the right problem? In *Findings of the Association for Computational Linguistics: ACL*, pp. 15537–15549, 2025.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning (ICML)*, 2020.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. VisionZip: Longer is better but not necessary in vision language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Xubing Ye, Yukang Gan, Xiaoke Huang, Yixiao Ge, Ying Shan, and Yansong Tang. VoCo-LLaMA: Towards vision compression with large language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Ao Zhang, Hao Fei, Yuan Yao, Wei Ji, Li Li, Zhiyuan Liu, and Tat-Seng Chua. VPGTrans: Transfer visual prompt generator across LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *International Conference on Machine Learning (ICML)*, 2025.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. In *International Conference on Learning Representations (ICLR)*, 2022.

Eric Zimmermann, Harley Wiltzer, Justin Szeto, David Alvarez-Melis, and Lester Mackey. KerJEPA: Kernel discrepancies for euclidean self-supervised learning. *arXiv preprint arXiv:2512.19605*, 2025.

A Limitations

This appendix follows the paper in order and states, for each claim, what the evidence does not support.

Every host in this paper is built on a frozen CLIP vision tower (Section 4), so every transfer we report crosses a decoder or a bridge on one encoder family; transfer across vision encoders (Oquab et al., 2024; Zhou et al., 2022) is untested. Every number is read at the epoch selected by a label-free rule, the minimizer of held-out loss on the 5% split, never at the labeled monitor's argmax. The ablation in Appendix B instead reports means over the last training epochs. The rule selects the final epoch on 43 of 48 arms, and on the main arm it gives up 0.9 to 2.9 monitor points against the run's own oracle. Only the main grid $K{=}144$ arm and the $K{=}72$ and $K{=}36$ ablation arms of Appendix B.3 are trained at three seeds. Every other arm, including the ladder, the $K{=}288$ cascade, the CLIP arms and the rest of the ablation, is a single seed, and its margins are point estimates the paper does not test.

The gain in Section 6.1 is measured against the parameter-free average pool rather than the untrained module, because the untrained grid module is an unseeded per-run draw and not a property of the geometry. It therefore measures the training stage as a whole, objective and learned map together.

Only the four episodic cells of Table 5 are paired; the other fourteen are corpus-level scores with no interval. The CLIP cell is measured inside the host that trained it and is not a transfer.

The transfer results of Section 5 carry five qualifications. The research host's 135M decoder sets a weak ceiling, 124.29 CIDEr on the monitor under greedy decoding against 133.3–145.6 for BLIP-family captioners (Li et al., 2022; 2023b), so retention against it is easier than the percentages suggest, and the host's question-answering rows are excluded for the same reason. The in-context margins on both generative hosts are tested against a three-seed dispersion, the only interval they carry; the paired per-question re-score on the deployment host was not run. Table 3 is a single-seed table, so none of its margins is tested for significance, and its TextVQA column runs without OCR tokens, so it compares modes within the table and matches no published TextVQA figure. The BLIP-2 host is our own reproduction rather than the released checkpoint, its $K{=}288$ cell is a bridge-isolation control, and its one-shot ceiling is too low to read. Retrieval is scored on our own seeded partition of Flickr30k rather than a Karpathy split, so its absolute scores do not compare with published ones, and its margins are within a point on a single seed.

Two further qualifications cut across the same section. The VQAv2, OK-VQA and POPE pools overlap the pretraining images, POPE at about 75%. Only our mode trains, so the overlap can only inflate our margins; the overlap-free redraw was not run for these trained compressors. Task structure and domain shift are also confounded: every advantage falls on the natural-photograph distribution the compressor was pretrained on, and every episodic deficit on fine-grained out-of-domain corpora. TextVQA separates capability from domain in one direction only; separating the other needs episodic corpora we do not have. Our configuration lists five such corpora, but only two were available.

The window of Section 6.2 is the attention-pool ladder, one arm per budget at one seed, read on the research-host episodes only. The main arm belongs to the grid family, whose ratio is fixed by the grid, so it is marked at its own budget and never traverses the ladder. On the deployment host the only point other than the main budget is the pool $K{=}288$ cascade of Table 2, a different geometry never budget-matched to the main arm, so where the window's edges fall on that host is not measured.

The two-term ablation of Section 6.3 (Appendix B) is one seed per cell, except the $K{=}72$ and 36 caption gaps, which carry three. Its E-600 comparisons use a conservative criterion. A difference counts only if it exceeds the two recipes' combined 95% half-width. Per-episode scores were not saved, so a paired test is not possible. The counts therefore understate how often one recipe leads. "Readable" is relative to a decoder: the 7B host reads pred-only's ladder tokens, so the pin is shown necessary for the 135M host and for the grid geometry, not for every decoder. The predictive term's caption gain depends on a content-bearing anchor. With the CLS top-k gather replaced by a convex anchor, pred + pin ties pin-only at $K{=}144$ (101.0 against 101.6) and collapses at $K{=}36$ (3.9 against 30.9), so the term refines a selection rather than replacing one; this registered test came out against the simplest form of the claim. The coordinate repair fails at $K{=}36$. On the grid at $K{=}144$, pin-only beats pred + pin on CUB 5-shot (80.6 against 78.0), the one significant reversal among the twelve 5-shot comparisons of the two pinned recipes. Neither content measure captures what a decoder can address: the probe measures reconstruction of the tower, and E-600 is a parameter-free mean-pool into a nearest-centroid classifier, so a recipe whose captions are unusable can still score well on both, as pred-only does. Whether label supervision on the compressor would move any boundary in this paper is not answered: the supervised arms completed and are excluded because their recipe carries an auxiliary term outside Eq. (8).

The latency figures in the conclusion are regime-dependent. The same analytic saving returns 8.4% of wall clock at batch 1 and 35.8% at batch 8 on the deployment host, so any figure quoted without its batch size misleads. The batch-1, 2 and 4 timings are archived runs of a module of identical architecture on the same device, not of the main arm's trained weights, admissible because latency depends on the operation graph and not on the weights. On the corrected cost ledger the parameter-free anchors are marginally cheaper than the trained arm, so the paper claims no cost advantage over them.

One trap of the apparatus is recorded for reuse. Two hosts built at the same training seed do not share a compressor initialization, because constructing the larger bridge advances the global random number generator before the compressor is built. Any comparison that fixes a seed and expects matched initialization across host builds is unsound on this code path.

B The Two Terms in Detail

The objective has two terms with separate jobs. The predictive term puts image content into the kept tokens, while the pin keeps these tokens on the frozen tower's manifold, in the coordinates that hosts

Table 6: The two terms, one row per recipe and budget. *Content*: probe-error reduction relative to the untrained gather ($\times 10^2$); for the grid, where no untrained probe was run, raw probe error (lower is better). *Coordinates*: held-out MMD to the tower's tokens ($\times 10^3$, last five epochs) and held-out token retention. r_{attn}: size of the attention branch of Eq. (4) relative to the anchor (absent on the grid; not ranked). *Research host*: C-MON window and ICL-0 CIDEr. E-600: 5-shot accuracy and its mean retention. *7B*: mean retention over the nine LLaVA-1.5-7B tasks. "Untrained" is each geometry's module at initialization: exactly the CLS top-k gather for attention pooling (Section 3.2) and a random map for the grid. One seed per cell; best trained recipe per column and budget in bold.

		Content	Coordinates and mixing			Research host		E-600, 5-shot			7B
K	recipe	probe gain	MMD	tok. ret.	r_{attn}	C-MON	ICL-0	CUB	Aircraft	ret. %	ret. %
Attention-pool ladder											
576	uncompressed	–	–	–	–	124.3	117.8	82.2	66.1	100.0	100.0
288	untrained	0	–	–	0	113.0	108.7	83.9	69.5	103.6	96.4
288	pred + pin	6.33	18.9	**0.99**	0.25	**117.0**	109.4	**85.1**	72.3	**106.4**	96.2
288	pin-only	0.13	**7.8**	**0.99**	0.23	116.4	**111.4**	84.7	70.1	104.5	96.7
288	pred-only	**6.87**	44.4	0.97	0.27	90.4	89.0	83.1	**72.9**	105.7	**96.8**
144	untrained	0	–	–	0	80.2	77.7	80.1	71.6	102.9	89.1
144	pred + pin	2.72	7.4	0.98	0.23	100.0	94.4	84.1	68.5	103.0	90.3
144	pin-only	0.31	**5.3**	0.98	0.19	**100.5**	**95.2**	82.4	68.3	101.8	90.6
144	pred-only	**3.37**	53.6	**0.99**	0.28	13.8	40.0	**85.3**	**77.8**	**110.7**	**91.0**
72	untrained	0	–	–	0	15.2	21.3	66.7	35.2	67.2	76.0
72	pred + pin	1.90	7.7	**0.93**	0.28	**68.1**	**68.8**	79.3	59.6	93.3	82.5
72	pin-only	0.47	**6.5**	0.90	0.26	65.9	66.3	76.6	56.0	88.9	81.7
72	pred-only	**2.43**	64.0	**0.93**	0.33	5.1	26.0	**85.0**	**73.0**	**106.9**	**83.8**
36	untrained	0	–	–	0	1.2	4.3	32.7	27.5	40.7	58.3
36	pred + pin	1.57	9.0	**0.69**	0.41	**31.7**	**34.5**	63.5	40.0	68.9	**70.9**
36	pin-only	0.39	**7.9**	0.61	0.35	22.2	25.5	54.1	36.3	60.4	67.3
36	pred-only	**3.00**	178.2	0.30	1.07	0.8	4.7	**71.6**	**51.6**	**82.6**	67.8
Grid geometry (content column: raw probe error, lower is better; K=144 only)											
144	untrained	–	–	–	–	–	–	62.4	43.9	71.2	78.2
144	pred + pin	**0.237**	5.5	**0.98**	–	106.2	99.1	78.0	62.2	94.5	**89.5**
144	pin-only	0.273	**4.7**	**0.98**	–	**106.3**	**100.3**	**80.6**	**63.5**	**97.0**	88.6
144	pred-only	0.251	253.1	0.14	–	0.0	0.2	52.2	33.6	57.2	45.4
36	untrained	–	–	–	–	–	–	55.6	36.0	61.0	52.8
36	pred + pin	–	5.0	**0.96**	–	**10.1**	**31.4**	**75.5**	**60.8**	**91.9**	**81.9**
36	pin-only	–	**4.1**	0.93	–	8.5	24.6	**75.5**	56.0	88.3	81.0
36	pred-only	–	166.8	0.14	–	0.1	0.1	61.2	44.3	70.8	47.3

built on the tower were fit to. This appendix gives the full ablation behind Section 6.3; Appendix A states what it does not show.

Measures and recipes. Both properties are read from the vision tower alone, without a decoder. Content is measured by an *information probe*: the compressor is frozen, and a fresh predictor is trained for eight epochs to rebuild all 576 tower tokens from the K tokens it emits. We report the held-out error reduction relative to the untrained gather at the same K, so 0 means no content beyond the selected patches. Coordinates are measured by the held-out MMD of Eq. (7) to the tower's tokens (mean of the last five epochs) and by held-out token retention.

We train three recipes for forty epochs: pred + pin (proposed), pin-only and pred-only. Each is trained on the attention-pool geometry at $K \in \{288, 144, 72, 36\}$ and on the grid at $K \in \{144, 36\}$, with one seed per cell unless stated otherwise. We then evaluate them with C-MON (research-host captions, mean of epochs 31–40), ICL-0 (the in-context harness at zero shots), E-600, and the nine zero-shot tasks of the deployment host (mean retention; Sections 5.1 and 5.2). Tie bands were fixed before the runs at 2σ of an earlier three-seed spread: 1.31 CIDEr on the C-MON window and 4.22 on ICL-0. The seed replicates in Appendix B.3 measure their own bands.

B.1 The predictive term puts content into the tokens

The horizontal axis of Figure 2a shows how much content each recipe keeps. Pin-only barely adds any. Its probe gain is only 0.1–0.5×10^{-2}, at most a fifth of pred-only's at every budget, and it clears the probe's noise band only at $K \leq 144$. Pred-only, in contrast, keeps the most (2.4–6.9×10^{-2}). Pred + pin retains 92, 81, 78 and 52% of that gain at K=288, 144, 72 and 36.

The gates of Eq. (4) show why. The relative size of the attention branch, r_{attn}, measures how much of each output token is mixed from across patches rather than selected. It is highest for pred-only at every K. In addition, pred + pin's value exceeds pin-only's at all four attention-pool budgets, and each paired 95% interval over 500 images excludes zero (Table 6). Thus, the predictive term is what opens cross-token mixing, whereas pin-only stays closest to plain selection.

Table 7: Which term each measure rewards. Numbers come from Table 6 and from the seed and dose follow-ups in Appendix B.3.

Measure	Uses	$K \geq 144$	$K \leq 72$
Information probe (tower only)	content	pred $\gg$ pin	pred $\gg$ pin
E-600 few-shot (pooled)	content	all above ceiling; pred-only best at 144	pred-only best; the pin costs 14
MMD, token retention (tower only)	coordinates	pin	pin
Captions, 135M decoder (sequence)	coordinates, then content	pin; pred adds 0	both; pred $+4.1$ / $+10.2$
LLaVA-1.5-7B, the other eight	either	all three within 0.7 on the mean	pred + pin leads $7/8$ at 72, $8/8$ at 36
TextVQA, LLaVA-1.5-7B	local detail	pred + pin trails pin-only by 2.6–2.9	trails by 3.6–5.4
Grid geometry, every task	coordinates	pin; pred-only collapses	pin; pred adds $+1.6$ / $+6.9$ on captions

Tasks that use this content directly follow the probe (Figure 2c). E-600 pools the K tokens into a class prototype, so no decoder stands between the tokens and the score. On this task, pred-only is the best recipe in 14 of the 16 attention-pool cells. It is still best in 12 when its lead over each other recipe must exceed their combined 95% half-widths.

Pred-only even beats the uncompressed model in every dataset–shot setting at K=144 and 72, with mean 5-shot retention of 110.7% and 106.9%. At K=36 it retains 82.6%, against 68.9% for pred + pin and 60.4% for pin-only. Adding the pin therefore costs 8–14 points of 5-shot retention at $K \leq 144$. This is the price of coordinates on a task that does not need them.

Nevertheless, the predictive term still helps on top of the pin. Pred + pin leads pin-only in all eight attention-pool 5-shot cells, and six of these leads exceed the combined half-widths. The leads include K=288 and 144, where the two recipes tie on captions, so the caption monitor misses this content.

B.2 The pin puts the tokens on the tower's manifold

The vertical axis of Figure 2a shows where the tokens sit. Without the pin, they drift. Pred-only's held-out MMD is 5.7–22.6× pin-only's across budgets, and its token retention falls to 0.30 at K=36. Pred + pin, on the other hand, stays within 1.1–2.4× of pin-only. The pin therefore holds the coordinates whether or not the predictive term is on.

Tasks that pass the tokens through the remaining frozen blocks and a decoder follow the MMD (Figure 2b). On the research host, pred-only reaches 73% of the caption ceiling at K=288. At the three tighter budgets, however, it reaches only 11, 4 and under 1%. It stays below the untrained gather at every budget. The two pinned recipes, by contrast, track each other and tie at K=288 and 144.

A *coordinate repair* separates the two properties directly. We take each final pred-only compressor and freeze everything that decides which patches are kept and how they are mixed. We then re-train only its per-token branch (the MLP branch of Eq. (4), with its gate and normalization) under the pin alone for ten epochs.

After the repair, captions (C-MON, mean of repair epochs 8–10) rise from 90.8 to 117.8 at K=288, from 14.2 to 104.5 at K=144, and from 5.2 to 70.7 at K=72. Each result exceeds pin-only's window after its full forty epochs (116.4, 100.5 and 65.9). The content was therefore already in the tokens, and only their coordinates were off. At K=36, however, the repair reaches 14.8 against pin-only's 22.2. At the tightest budget, the pin therefore does more than fix coordinates.

The grid shows the pin's job in isolation. Because grid pooling makes no selection, the predictive term has no anchor to refine. As a result, without the pin the tokens leave the manifold entirely. At K=144, pred-only's MMD is 0.25, against 0.005 with the pin. Its token retention is 0.14, its research-host captions are 0.0, and its deployment-host mean retention is 45%.

Moreover, pred-only's probe error (0.251) is higher than pred + pin's (0.237), so part of the content is lost off the manifold as well. This explains why pred-only keeps the most content with attention pooling but not on the grid. The reversal comes from this mechanism, not from a change in what either term does.

B.3 Where each term pays

Table 7 groups the measures by the property they reward. The pattern holds at every budget, and only the size of each effect changes with K.

Tasks that use content directly want the predictive term and pay for the pin. On the probe and on E-600, pred-only leads at every attention-pool budget from K=144 down. Adding the pin gives up some of that content for coordinates these tasks never use.

Sequence tasks on the research host need the pin first. Without it, captions fall below the untrained gather at every budget. With it, the predictive term adds nothing at $K \geq 144$: $+0.5$ and -0.5 at K=288 and 144, both inside the tie band. At K=72 and 36, however, it adds $+2.1$ and $+9.5$.

With two more seeds, these gaps average $+4.08$ and $+10.19$, against noise bands of 2.25 and 2.46 measured on the same seeds. Pred + pin is also ahead in every seed pair on both the window and ICL-0.

The $K{=}36$ gap grows with the predictive weight. With the pin on throughout, captions reach 22.2, 27.3, 31.2, 31.7 and 33.0 CIDEr at $w_{\text{pred}} = 0$, 0.25, 0.5, 1 and 2. Over this range, the held-out MMD stays between 0.0079 and 0.0099. The gain therefore comes from the predictive term itself, not from better readability.

The grid shows the same crossover. Pred + pin ties pin-only at $K{=}144$ (-0.1 on the window, -1.3 on ICL-0) and leads at $K{=}36$ ($+1.6$ and $+6.9$). In short, once selection or pooling no longer keeps enough of the image, content becomes the limiting factor, and the predictive term supplies it.

The deployment decoder barely separates the recipes until the budget is tight. At $K \geq 72$, LLaVA-1.5-7B reads all three recipes about equally. Mean retention over the nine tasks differs by at most 0.7 points at $K \geq 144$ and by 2.2 at $K{=}72$. It also reads pred-only's off-manifold attention-pool tokens as well as pin-only's. Readability thus depends on the decoder as well as on the tokens: a 7B decoder with a trained projector tolerates a drift that the 135M host cannot.

The recipes separate only at $K{=}36$, where pred + pin leads pin-only on eight of the nine tasks: $+11.9$ on COCO captioning, $+5.5$ on OK-VQA, $+4.5$ on VQAv2, and $+3.5$ on the mean. The exception is TextVQA, where pred + pin trails pin-only at every attention-pool budget, by 2.6–5.4 points. This is consistent with the predictive term's cross-token mixing blurring the local detail that reading needs. Section 5.2 reports the same boundary.

B.4 Why the decomposition is useful

Reading the objective as content plus coordinates, rather than as a single score, has four benefits.

Failures become easy to diagnose. Both measures come from the frozen tower alone, with no labels and no decoder. A compressor that fails on a host can therefore be diagnosed in one pass. A small probe gain means the tokens do not carry the image, so the predictive term needs more weight. A large MMD means they carry it in coordinates the host cannot read, so the pin needs more weight. The coordinate repair in Appendix B.2 fixes the second case in ten epochs.

The boundaries in Sections 5 and 6.2 gain a mechanism. The window narrows at $K{=}288$ because the gather already keeps the content: its captions reach 91% of the ceiling before training, and the pin alone does the rest. A learned rule has little to gain where the fixed one loses little. On the same view, the window closes on TextVQA because the predictive term gathers summarized content, whereas reading needs the unsummarized patch.

The recipe can be chosen by task. With attention pooling, a host that pools the tokens, like the prototype classifier here, wants the predictive term and can lighten or drop the pin. Indeed, pred-only beats the uncompressed model on fine-grained few-shot recognition at a quarter of the tokens. A host that reads the tokens as a sequence through a small decoder, however, needs the pin. At an eighth of the tokens or fewer, it needs both terms.

The proposed recipe is a compromise between these two cases. On sequence tasks, it stays within 2.0 CIDEr of the better single term at every budget. On pooled tasks, it gives up 8–14 retention points to pred-only at $K \leq 144$. Each single term, by contrast, loses by 18 retention points or more on some task: pred-only on captions, and pin-only on few-shot recognition at $K \leq 72$. On the grid, which makes no selection, neither kind of task can drop the pin.

The decomposition also explains why the compressor transfers. The pin ties the tokens to the frozen tower's own distribution. Since every host in this paper is built on that tower, tokens that satisfy the pin are in coordinates that all of these hosts were fit to. The cross-bridge transfer of Section 5.1 relies on this property. The predictive term, in turn, supplies content with no host in the loop. Finally, because neither term ever sees a label or a decoder, the compressor does not depend on any particular host.