DATASET-LEVEL REGULARIZATION FOR JOINT-EMBEDDING SELF-SUPERVISED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Self-supervised joint embedding learning aligns representations of augmented views while regularizing their distribution to prevent collapse. Regularization aims to shape the overall representation distribution, but most joint embedding methods restrict its computation to the active minibatch. We propose out-of-batch regularization, which maintains a dataset-level state in CPU memory and uses globally constructed per-instance targets during minibatch training. Evaluated on natural images and biomedical datasets, these anchors approximate the full-dataset regularization gradient more faithfully than conventional minibatch estimation at small batch sizes. Applications to contrastive (SimCLR) and non-contrastive (SIGReg) approaches show consistent improvement over the corresponding in-batch objectives for image classification, minority-class performance under controlled imbalance, instance retrieval, and continued self-supervised pretraining, with the largest gains at small batch sizes. These results suggest that dataset-level regularization can be decoupled from minibatch optimization, making batch size an optimization choice that does not affect the geometry of the regularizer.

1 INTRODUCTION

Joint-embedding architectures (JEAs) (Chen et al., 2020a; Grill et al., 2020; Caron et al., 2021; Bardes et al., 2021) have been the dominant paradigm for visual self-supervised representation learning. Applications range from 3D medical image analysis (Tang et al., 2022) to 3D scene reconstruction (Wang et al., 2025) and vision–language alignment (Radford et al., 2021). They use augmentations to create a pretraining objective based on view invariance. Anti-collapse regularization complements this objective to prevent trivial solutions (Chen & He, 2021). While different regularization mechanisms have been proposed over the years, most of them act on the embeddings of the current minibatch (or a few recent batches with EMA) that might be too small to represent the dataset under memory or optimization constraints.

The bottleneck is particularly relevant for high-dimensional inputs, including high-resolution images and 3D volumes, where each sample requires more activation memory. Such applications are thus often limited to small-batch training. The problem worsens for imbalanced data, where rare classes may contribute only a few examples to each batch (Jing et al., 2021). The simple solution of using a larger batch is constrained by hardware (Gao et al., 2021), and even when a large batch is possible, it may not be ideal for optimization. On small datasets, using a large fraction of the training samples per batch leaves few parameter updates per epoch. Masters & Luschi (2018) also report that larger batches can require more careful choice of the learning rate and training duration. The batch size needed to estimate a population regularizer can therefore differ from the best batch size for optimization.

We propose a framework that moves the statistical context of SSL regularizers from the active batch to the dataset-level state. We maintain a detached, dataset-indexed state in CPU memory. A periodic refresh encodes the training set and constructs globally optimized targets, while online steps retrieve only the current samples' targets (Fig. 1). Thus, the support of the anti-collapse component is no longer limited by the batch size or GPU capacity.

Our main contributions are:

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.

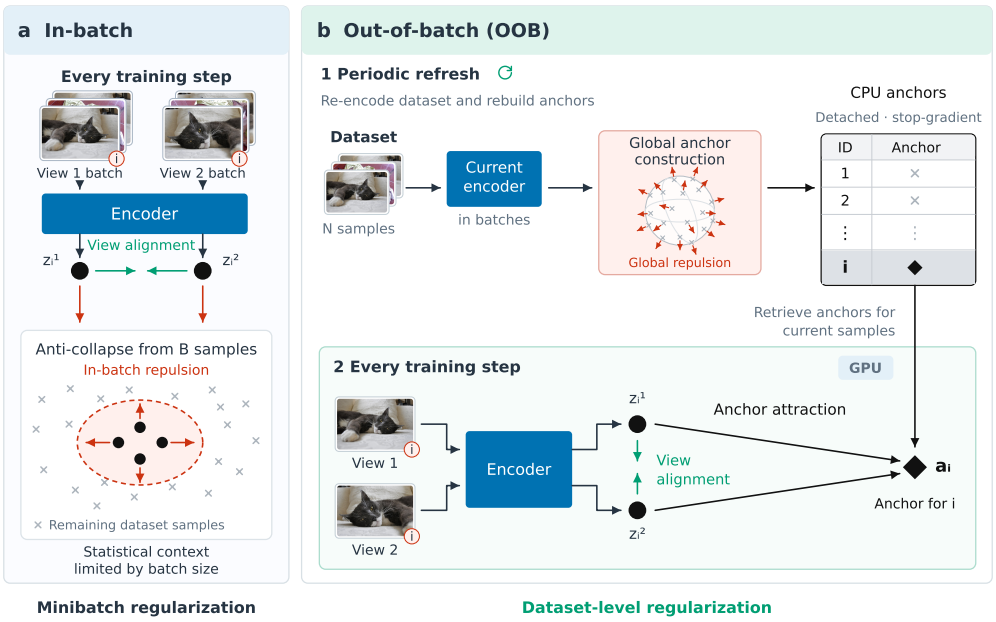

Figure 1: **Minibatch and OOB regularization.** (a) In-batch methods estimate the anti-collapse term from active samples at each step. (b) OOB regularization periodically encodes the dataset, constructs detached per-instance anchors, and retrieves the anchors indexed by the online minibatch.

- We introduce OOB regularization, where offline, dataset-level targets act as per-instance anchors during online training, and derive contrastive (CONTRASTIVE-OOB) and non-contrastive (SIGREG-OOB) instances of this construction.
- We formulate anti-collapse as estimation of a finite population gradient and measure minibatch error and cache staleness on 19 real datasets. The analysis also separates gradient-estimation error from population reweighting under class imbalance.
- Across natural images and 2D/3D biomedical data, the OOB variants yield positive mean gains over their matched in-batch objectives in image classification evaluated by kNN and linear probes. The gains are largest at small batch sizes and extend to minority-class recall, instance retrieval, and continued pretraining.

2 RELATED WORK

Collapse prevention in joint embedding training. Joint-embedding methods must preserve informative variation while aligning views. Contrastive objectives achieve this through negative-sample repulsion (Oord et al., 2018; Chen et al., 2020a), linking alignment to hyperspherical uniformity for normalized representations (Wang & Isola, 2020). To avoid explicit negatives, Barlow Twins and VICReg constrain correlations and moments (Zbontar et al., 2021; Bardes et al., 2021), while SIGReg matches projected distributions to a Gaussian (Balestriero & LeCun, 2025). Other approaches prevent collapse through balanced assignments in SwAV (Caron et al., 2020) or asymmetry, teachers, and stop-gradient in BYOL, DINO, and SimSiam (Grill et al., 2020; Caron et al., 2021; Chen & He, 2021). These approaches establish how to prevent collapse, but they have to assume that the active batch is representative of the dataset.

Cross-batch state and global objectives. Reusing representations across updates expands batch context. MoCo's momentum-encoder queue (He et al., 2020; Chen et al., 2020b), instance discrimination's memory bank (Wu et al., 2018), and cross-batch memory (Wang et al., 2020) expand the pool of negative samples. NNCLR uses stored embeddings to find positive neighbors (Dwibedi et al., 2021). Chen et al. (2022) argues that minibatch contrastive gradients can be biased, motivating global-objective optimization through per-image moving averages in SogCLR (Yuan et al.,

2022) and MCMC negative sampling in EMC^2 (Yau et al., 2024). Decoupled Contrastive Learning reduces batch sensitivity by removing positives from the denominator (Yeh et al., 2022). OOB instead performs population regularization periodically and retrieves per-instance targets for online attraction, avoiding cross-instance comparisons in the online regularizer.

Assigned targets and dataset-indexed representations. Constructing targets separately from encoder updates also provides broader context. Noise-vector assignment (Bojanowski & Joulin, 2017), dataset-feature clustering (Caron et al., 2018), and balanced self-labeling through optimal transport (Asano et al., 2019) organize representation learning around assigned targets. Our anchors instead retain per-instance correspondence and are reconstructed from current representations under a specified continuous geometry. DIET also uses sample identity, predicting datum indices as classes (Balestriero, 2023). IConE, the conceptually closest recent approach (Almpanakis & Kreshuk, 2026), separates collapse prevention from encoder alignment through a jointly optimized $N \times d$ table of auxiliary instance embeddings. Its regularization maintains a persistent global parameter space during training and evaluates an $N \times N$ similarity matrix each step. OOB instead periodically constructs detached per-instance targets from a chosen population-level regularizer, stores these targets in CPU memory, and retrieves only those required by the active minibatch. This decouples online training from the global computation while supporting different regularizers.

3 METHOD

To give small-batch training access to dataset-wide regularization, we construct anti-collapse anchors from the full dataset and use them to update the encoder. We first formulate this separation, instantiate it with spherical (contrastive) and Gaussian (non-contrastive) anchors, and then define a full-dataset gradient reference for comparing minibatch estimation with cached regularization.

3.1 ALIGNMENT AND POPULATION REGULARIZATION

We use mutual information as an organizing view of alignment and anti-collapse, without assuming that joint-embedding methods directly optimize it. For representations Z_1 and Z_2 of related views, the mutual information (MI) identity (Shannon, 1948) gives

$$I(Z_1; Z_2) = H(Z_1) - H(Z_1 \mid Z_2). \tag{1}$$

This identity motivates two complementary roles in JEA losses: alignment reduces uncertainty between views, while anti-collapse keeps the marginal representation informative. Representation quality nevertheless depends on more than MI (Tschannen et al., 2019).

Let $\mathcal{D} = \{x_i\}_{i=1}^N$ be a dataset of unlabeled inputs, and let f_θ map an input to its representation. For a sample x drawn uniformly from $\mathcal{D}$, let t_1, t_2 be independently sampled transformations from an augmentation family $\mathcal{T}$, producing views $t_1(x)$ and $t_2(x)$. A broad JE-SSL objective is

$$\mathcal{L}_{\text{SSL-JE}}(\theta) = \underbrace{\mathbb{E}_{x,t_1,t_2}[\mathcal{L}_{\text{align}}(f_\theta(t_1(x)), f_\theta(t_2(x)))]}_{\text{Alignment}} + \underbrace{\lambda \mathcal{R}(P_\theta)}_{\text{Anti-collapse}}, \tag{2}$$

where P_θ is the distribution of $f_\theta(t(x))$ induced by sampling inputs and augmentations.

In practice, a regularizer sees a finite set of representations. Writing $z_i = f_\theta(t(x_i))$, the empirical distributions of an active minibatch $\mathcal{B}$ of size B and the full training set are

$$\widehat{P}_B = \frac{1}{B} \sum_{i \in \mathcal{B}} \delta_{z_i}, \qquad \widehat{P}_N = \frac{1}{N} \sum_{i=1}^{N} \delta_{z_i}. \tag{3}$$

Here δ_{z_i} is a unit point mass at z_i, giving each included representation equal probability. Standard in-batch training evaluates $\mathcal{R}(\widehat{P}_B)$ in place of the population term in eq. (2). Our aim is to supply online updates with the broader context of $\mathcal{R}(\widehat{P}_N)$ without recomputing it through the encoder at every step. We therefore construct dataset-level anchors periodically and use a separable sample-to-anchor loss between refreshes.

3.2 A REFRESH-AND-TRAIN INTERFACE

Let $z_i^{(v)}$ denote the representation of augmented view v of sample i. Our OOB methods optimize

$$\mathcal{L}_{\text{OOB}} = \lambda_{\text{align}}\mathcal{L}_{\text{align}} + \lambda_{\text{ac}}\mathcal{L}_{\text{ac}}^{\text{OOB}}, \tag{4}$$

where $\mathcal{L}_{\text{align}}$ maximizes view invariance and $\mathcal{L}_{\text{ac}}^{\text{OOB}}$ attracts each representation to its dataset-indexed anchor. To implement $\mathcal{L}_{\text{align}}$, we maximize cosine similarity between embeddings produced by different views of the same instance. To compute $\mathcal{L}_{\text{ac}}^{\text{OOB}}$, at refresh r, the current encoder processes one view of every training sample. A population operation (sections 3.3 and 3.4) converts outputs into detached anchors $A^{(r)} = \{a_i^{(r)}\}_{i=1}^N$, which are stored in CPU memory. For an online minibatch $\mathcal{B}$, only its indexed rows are moved to the GPU (Table 1):

$$\mathcal{L}_{\text{ac}}^{\text{OOB}} = \frac{1}{BV}\sum_{i\in\mathcal{B}}\sum_{v=1}^{V} d_{\mathcal{G}}\left(h_r(z_i^{(v)}), a_i^{(r)}\right). \tag{5}$$

Here $d_{\mathcal{G}}$ measures discrepancy from the anchor using MSE across projections for Gaussian anchors, or cosine distance for L_2-normalized spherical anchors; h_r applies refresh-fixed projections (Gaussian anchors) or L_2 normalization (spherical anchors). Gradients pass through z_i, but not through the anchors, and the online anti-collapse term requires no relationship among samples in $\mathcal{B}$.

Table 1: OOB training separates global target construction from online parameter updates.

	Refresh phase
1	Run inference in minibatches over the entire dataset, collecting one view representation per sample.
2	After the full inference pass, construct $a_i^{(r)}$ jointly from all N representations.
3	Detach $A^{(r)}$ and store it by sample index in CPU memory.
	Online phase, repeated until the next refresh
4	Sample $\mathcal{B}$ and generate V augmented views per sample.
5	Retrieve $\{a_i^{(r)} : i \in \mathcal{B}\}$ and evaluate eq. (4).
6	Update encoder and projector parameters; keep anchors fixed.

3.3 SIGREG-OOB: PROJECTED GAUSSIAN ANCHORS

SIGREG-OOB constructs a Gaussian target through one-dimensional rank transport assignment to a fixed Gaussian quantile grid (Gabriel & Marco, 2019). At refresh, choose unit directions $v_1, \ldots, v_M$. For $s_{im} = v_m^\top z_i$, sort all N samples and assign the rank-k sample to

$$\gamma_k = \Phi^{-1}\left(\frac{k+1/2}{N}\right), \qquad k = 0, \ldots, N-1. \tag{6}$$

If r_{im} is the rank of sample i along m, its cached coordinate is $q_{im} = \gamma_{r_{im}}$. The online term is

$$\mathcal{L}_{\text{ac}}^{\text{SIGReg-OOB}} = \frac{1}{BVM}\sum_{i\in\mathcal{B}}\sum_{v=1}^{V}\sum_{m=1}^{M}\left(v_m^\top z_i^{(v)} - q_{im}\right)^2. \tag{7}$$

Refresh inference uses GPU, while the M sorts and the $N \times M$ target table are computed on CPU. We resample the projection directions at each refresh. Our SIGREG baseline uses the same sorted-quantile targets computed in the batch, rather than the Epps-Pulley statistic (Balestriero & LeCun, 2025), so we keep the same regularizer. We ablate SIGREG-OOB hyperparameters in section A.4.

3.4 CONTRASTIVE-OOB: SPHERICAL ANCHORS

For contrastive learning, anchors lie on the unit sphere. Starting from the normalized refresh representations $\bar{z}_i^{(r)}$, we define the population target by

$$\mathcal{U}(A) = \log\left[\frac{1}{N(N-1)}\sum_{i\neq j} e^{-\tau_u\|a_i - a_j\|_2^2}\right], \tag{8}$$

$$A^{(r)} \approx \arg\min_{A\in(\mathbb{S}^{d-1})^N} \mathcal{U}(A) + \frac{\lambda_f}{N}\sum_i\left(1 - a_i^\top \bar{z}_i^{(r)}\right). \tag{9}$$

The first term favors a globally spread spherical population (Wang & Isola, 2020); the second preserves anchor–sample correspondence. With $u_i^{(v)} = z_i^{(v)}/\|z_i^{(v)}\|_2$, online training uses

$$\mathcal{L}_{\text{ac}}^{\text{Contrastive-OOB}} = \frac{1}{BV}\sum_{i\in\mathcal{B}}\sum_{v=1}^{V}\left(1 - {u_i^{(v)}}^{\top} a_i^{(r)}\right). \tag{10}$$

We optimize Eq. 9 stochastically, avoiding all $N(N-1)$ pairs. Each of 20 steps evaluates all pairs among up to 256 sampled anchors. The first refresh initializes from the source plus small Gaussian noise. SimCLR obtains spherical spreading from minibatch negatives, while MoCoV2 extends its negative support with a momentum queue. CONTRASTIVE-OOB instead performs repulsion during anchor construction and uses only sample-to-anchor attraction online, without in-batch negatives or a momentum encoder. This preserves the intended spherical spreading geometry that contrastive learning imposes. For specific training settings, see section A.3.

3.5 COMPARING MINIBATCH AND CACHED GRADIENTS

Having defined the training methods, we compare a minibatch estimate and a cached estimate against the same full-dataset gradient at the current encoder state. Differential entropy is a property of the underlying distribution, but training provides only a finite set of encoder outputs. Practical methods optimize finite-sample surrogates that spread or decorrelate these outputs. Using the full empirical distribution in eq. (3), the reference for a chosen surrogate is its full empirical gradient

$$g_i^\star = \nabla_{z_i}\mathcal{R}(\widehat{P}_N). \tag{11}$$

We call $g_i^\star$ the oracle gradient. It is the representation-space direction supplied by the chosen regularization when that surrogate sees the complete empirical population. Minibatch gradient bias for contrastive objectives has been established in prior work (Chen et al., 2022; Yuan et al., 2022), but here we compare representation-space gradients for a fixed finite-population regularizer. The standard minibatch estimator substitutes the active empirical distribution $\widehat{P}_B$:

$$\widehat{g}_{i,B} = \nabla_{z_i}\mathcal{R}(\widehat{P}_B). \tag{12}$$

An OOB estimator instead constructs dataset state $\mathcal{C}_{t_0} = \{c_i\}_{i=1}^N$ at refresh step t_0 and later uses

$$\widetilde{g}_{i,t} = \nabla_{z_i}\mathcal{L}_{\text{ac}}(z_i, c_i), \qquad t \geq t_0. \tag{13}$$

We measure directional agreement of cached and minibatch approximations with the oracle using cosine similarity and scale-sensitive error using rRMSE.

Computing the reference and its estimators. All three estimators differentiate the same error between projected representations and Gaussian quantile targets (SIGReg, SIGREG-OOB). The oracle ranks the full current dataset, the minibatch estimator ranks each probe within sampled batches of size B, and the cached estimator reuses full-dataset assignments from refresh step t_0. At cache age s, we recompute the oracle using the current encoder at $t_0 + s$, so the comparison measures staleness against the current population. Experimental results are in section 4.2.

Separating estimation from population weighting. Class imbalance also leads to the desired representation prior not matching the data's class proportions (Assran et al., 2022). If class c has N_c samples, the empirical population weights it by N_c/N. To separate estimator error from the weighting, we sample another population in a label-aware manner for evaluation:

$$\widehat{P}_{\text{bal}} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i:y_i=c}\delta_{z_i}. \tag{14}$$

Comparing its gradient with the empirical-population gradient separates a change in the objective's weighting from an error in estimating a fixed objective. Gradient comparisons and downstream imbalance results appear in section 4.4.

4 RESULTS

We examine gradient-estimation accuracy (section 4.2), frozen classification performance across batch sizes (section 4.3), population weighting and minority-class recall under imbalance (section 4.4), instance retrieval (section 4.5), adaptation of pretrained encoders (section 4.6), and performance under GPU-memory constraints (section 4.7).

4.1 DATASETS

We experiment on natural and scientific/biomedical images: CIFAR-10/100 (Krizhevsky et al., 2009), DTD (Cimpoi et al., 2014), STL-10 (Coates et al., 2011), ImageNet-100, Imagenette, and Imagewoof (Deng et al., 2009); BBBC010 (Ljosa et al., 2012), twelve 2D and six 3D MedMNIST datasets (Yang et al., 2023), and a private plankton dataset imaged with widefield microscopy. Here we present aggregate results, see section C for complete dataset-level results and size sweeps.

4.2 ORACLE GRADIENT ANALYSIS

Using the fixed objective from section 3.5, we compare minibatch sampling error with cache staleness at training checkpoints. Figure 2A–B shows that limited batch context can change the direction of regularization and can produce gradients that oppose the full-dataset objective. As batch size gets bigger, its regularization gradient agrees more with the oracle. A fresh cache recovers the oracle exactly, but its agreement deteriorates as the encoder changes (Figure 2C–D). This deterioration is slower at later checkpoints, where cached gradients remain informative across more online updates. The results confirm that minibatch and cached estimators face different limitations: incomplete population context versus temporal staleness. When training with small batches, periodic dataset-level regularization is clearly beneficial, while the update schedule should depend on the training stage.

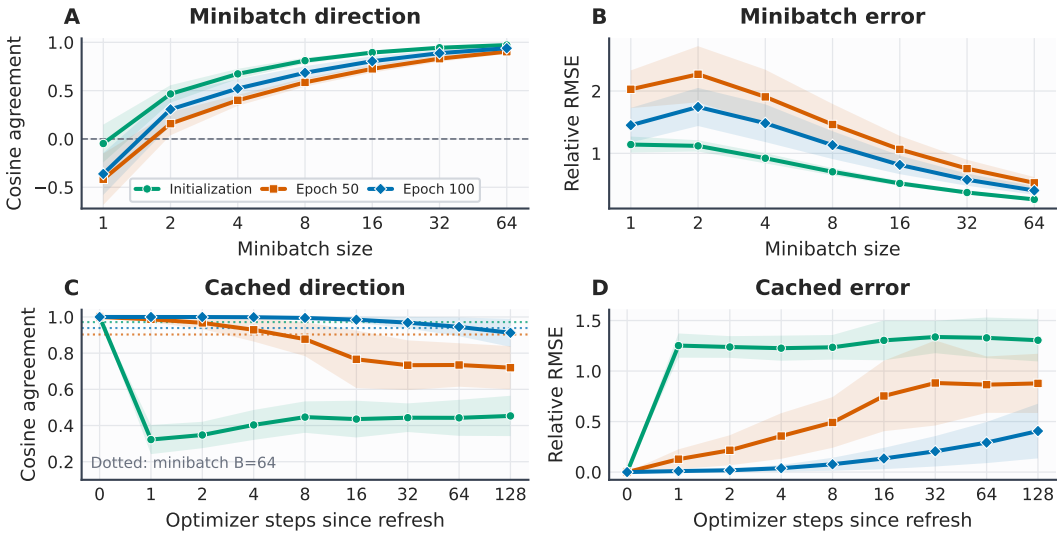

Figure 2: Minibatch and cached gradient agreement with the full-dataset reference.

4.3 IMAGE CLASSIFICATION ACROSS BATCH SIZES

Evaluation protocol. We compare each OOB method with its corresponding baseline: SIGREG-OOB with SIGREG, and CONTRASTIVE-OOB with SimCLR and MoCoV2. Within a comparison, we keep the data split, augmentations, number of views, backbone, projector, optimizer, learning rate schedule, and evaluation fixed. The 2D experiments use a ViT backbone (Dosovitskiy et al., 2020); 3D experiments use a 3D ResNet (He et al., 2016) and training lasts 100 epochs. Frozen representations are evaluated by k-NN and linear probe (Wu et al., 2018; Chen et al., 2020a). We use three independent seeds. In-batch baselines are omitted at $B = 1$, because anti-collapse regularization based on a single-sample batch is either undefined or statistically uninformative. OOB methods encode dataset-level state and remain well defined. Training details are in section A.2.

Results. The low-batch regime shows the clearest separation (Figure 3). Averaged over small batch sizes, SIGREG-OOB improves 5-NN/linear probe performance over SIGREG by $14.17/12.89$pp on natural images, while CONTRASTIVE-OOB improves over SimCLR by $10.83/6.58$pp on biomedical data. On 3D volumes, both OOB methods retain positive mean gains, showing that the effect is not tied to a 2D encoder. Complete paired aggregates are in Table 2. Section C adds compar-

isons with other baselines like IConE, BYOL, DINO, SwaV and VICReg. Section C.1 contains representation metrics like RankMe and effective-rank (Garrido et al., 2023; Roy & Vetterli, 2007).

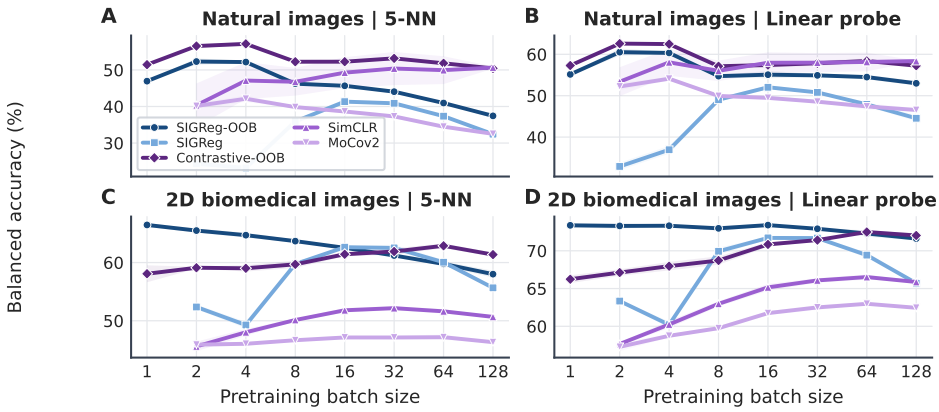

Figure 3: Image classification results using linear probes and 5-NN.

Table 2: Classification balanced accuracy (%). Parentheses show OOB gains in percentage points.

	SIGReg		SIGReg-OOB (Ours)		SimCLR		Contrastive-OOB (Ours)	
Benchmark	5-NN	Linear	5-NN	Linear	5-NN	Linear	5-NN	Linear
Natural images	34.63	45.55	48.80 (+14.17)	58.44 (+12.89)	50.21	59.35	52.50 (+2.29)	59.49 (+0.14)
2D biomedical	57.45	67.44	62.21 (+4.76)	72.84 (+5.40)	50.00	63.55	60.83 (+10.83)	70.13 (+6.58)
3D biomedical	51.39	55.89	55.61 (+4.22)	63.95 (+8.06)	51.76	57.52	55.47 (+3.71)	61.78 (+4.26)

4.4 Population weighting and minority classes under imbalance

Oracle diagnostic. Class-balanced reference eq. (14) is used to distinguish estimator error from the choice of population weighting. The exact empirical gradient for minority probes diverges from the class-balanced gradient as minority classes are downsampled (fig. 4). A cached empirical estimator tracks the empirical objective closely and a cache built with class-balanced targets tracks the class-balanced oracle. OOB state solves the estimation problem, but not class imbalance. When class information is available, population weighting can be used as a hyperparameter.

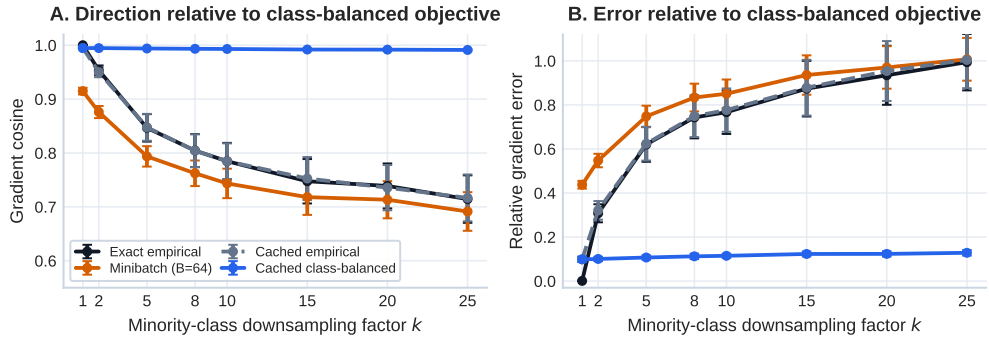

Figure 4: Empirical and class-balanced gradient agreement under class imbalance.

Downstream evaluation. Having separated these effects, we test whether OOB anchors improve minority-class representations without using labels during training. We create imbalanced MedMNIST variants by downsampling some classes by k. We report macro-average recall over downsampled classes using the same held-out split for matched methods. Across different batch sizes, SIGReg-OOB gains 6.24 points in 5-NN minority recall and 8.6 points in linear minority recall

over SIGREG. CONTRASTIVE-OOB gains 10.69 and 10.28 points over SimCLR. The gains remain positive across the downsampling sweep (Figure 5). These downstream results complement Figure 4: an empirical OOB cache does not rebalance the population, but it gives each rare sample a global context that is often absent from its minibatch. Full 2D/3D per-dataset results, worst-class recall, and minority–majority gaps are in section D.

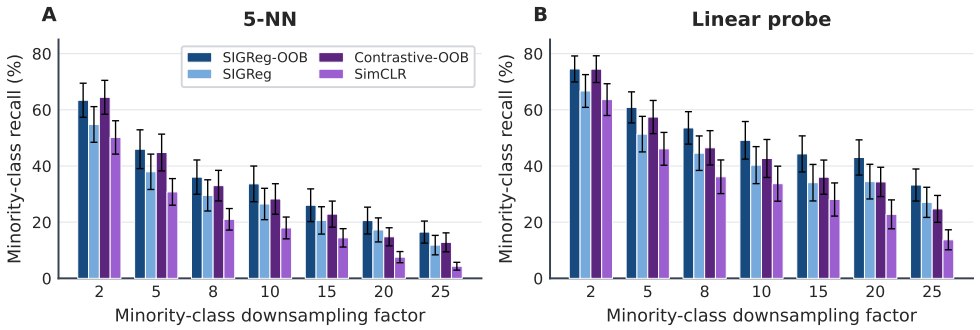

Figure 5: Minority-class recall under controlled class imbalance.

4.5 INSTANCE RETRIEVAL

We next test whether the learned space preserves instance neighborhoods in addition to the class boundaries. After pretraining, we retrieve with cosine similarity between frozen features. We report mean average precision (mAP) over three seeds and show the four nearest neighbors of the same query by each method (fig. 6). Across our experiments, SIGREG-OOB improves mAP over SIGREG by 13.1 points. CONTRASTIVE-OOB improves over SimCLR/MoCoV2 by 13.7/7.6 points, at a given batch size. Recall@1/5/10 and more retrieval evaluations are in section E.

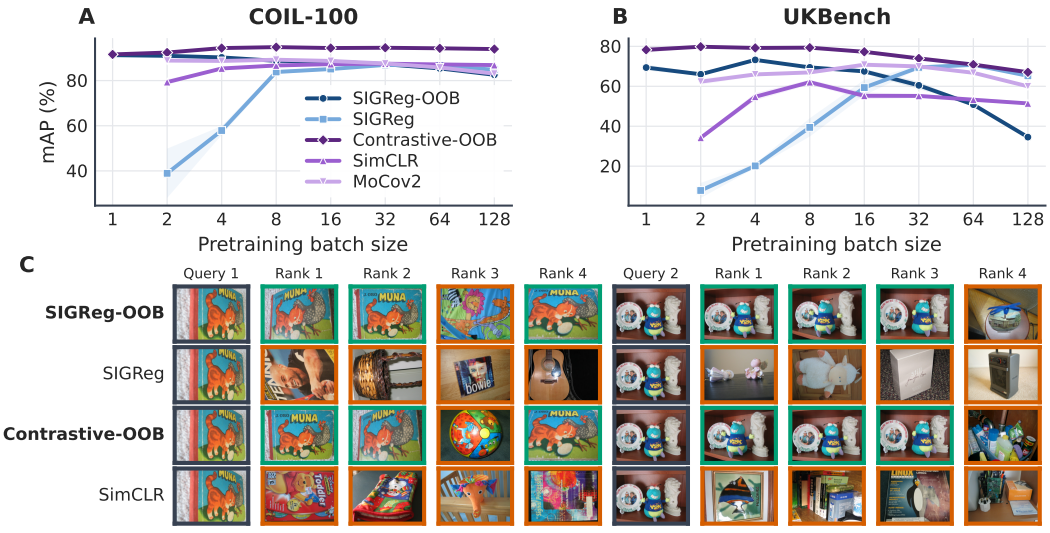

Figure 6: Instance-retrieval mAP and example nearest neighbors.

4.6 CONTINUED SELF-SUPERVISED PRETRAINING

Continued SSL pretraining adapts an existing model to a target domain (Reed et al., 2022; Rodas et al., 2025). Small batches can make a larger pretrained encoder feasible during domain adaptation. We train DINOv2 (Oquab et al., 2023) of several sizes on our datasets, without labels.

Table 3 shows that the OOB methods outperform their corresponding in-batch variants across most configurations. The advantage is clearest at small batches, where OOB regularization improves

Table 3: Continued pretraining of DINOv2 and 5-NN of adapted models relative to original.

			SIGReg-OOB (Ours)			SIGReg			Contrastive-OOB (Ours)			SimCLR		
Dataset	Encoder	Original	$B=2$	$B=8$	$B=64$	$B=2$	$B=8$	$B=64$	$B=2$	$B=8$	$B=64$	$B=2$	$B=8$	$B=64$
BloodMNIST	S (22M)	92.0	**+6.6**	**+6.7**	**+6.8**	-6.5	**+6.7**	+6.0	**+6.1**	+6.2	+5.8	+1.6	**+6.7**	**+6.5**
	B (86M)	90.1	**+8.8**	+8.6	**+8.6**	-5.7	**+8.7**	+8.4	**+8.0**	+8.4	+8.1	+4.1	**+8.7**	**+8.7**
	L (304M)	89.4	**+9.3**	**+9.4**	**+9.3**	-4.6	**+9.4**	+9.0	**+9.1**	+8.9	**+9.5**	+2.5	**+9.4**	+9.2
	G (1.1B)	93.0	**+5.3**	+5.1	**+5.8**	-6.8	**+5.8**	+5.3	+2.8	-5.6	+3.7	**+3.6**	**+5.5**	**+5.7**
OrganAMNIST	S (22M)	85.7	**+7.1**	+6.1	**+5.9**	+0.9	**+8.0**	+4.1	**+8.7**	**+8.2**	**+8.1**	-15.9	+4.6	+4.7
	B (86M)	84.0	**+10.4**	+8.6	**+7.1**	-0.4	**+8.7**	+4.6	**+10.6**	**+10.5**	**+10.3**	-18.0	+7.1	+6.3
	L (304M)	79.3	**+15.1**	**+14.3**	**+12.0**	+0.8	+13.0	+9.5	**+14.0**	**+15.2**	**+15.6**	-15.0	+10.9	+10.5
	G (1.1B)	80.0	**+14.0**	**+13.1**	**+11.6**	+10.1	+12.3	+9.6	**+13.5**	+11.2	+11.9	-8.9	**+11.3**	**+12.1**
OrganCMNIST	S (22M)	81.2	**+1.7**	**+4.5**	**+4.0**	-5.9	+4.1	+2.7	**+6.8**	**+6.8**	**+4.7**	-12.2	+3.2	+0.9
	B (86M)	77.3	**+6.5**	**+7.7**	**+6.6**	-0.1	+7.5	+6.0	**+9.3**	**+11.9**	**+9.1**	-16.5	+7.3	+5.1
	L (304M)	76.3	**+8.3**	**+7.9**	**+8.3**	-0.2	+7.7	+5.6	**+10.1**	**+13.9**	**+12.3**	-7.0	+7.2	+4.9
	G (1.1B)	74.1	**+11.1**	**+10.6**	**+8.8**	+10.7	**+10.6**	+7.7	**+13.4**	**+12.0**	**+12.3**	-3.2	+11.3	+6.6
PathMNIST	S (22M)	94.8	**+2.5**	**+2.6**	**+2.9**	-7.6	+1.0	+1.9	**+1.3**	**+2.2**	**+2.1**	-9.4	+1.8	+1.9
	B (86M)	94.0	**+3.5**	**+3.6**	**+3.6**	-9.6	+2.5	+2.6	**+2.2**	+2.9	+2.7	-6.0	**+3.2**	**+3.6**
	L (304M)	93.7	**+4.0**	**+3.2**	**+4.6**	-14.1	+3.0	+3.3	**+2.3**	+2.3	+3.1	-3.7	**+3.8**	**+3.8**
	G (1.1B)	95.5	**+1.5**	**+2.2**	**+2.2**	-13.9	+0.9	+1.2	**-1.5**	-6.7	-1.8	-2.3	**+1.6**	**+2.3**

on the pretrained initialization while in-batch training can degrade representation quality. These results support using OOB regularization to adapt large pretrained encoders when memory limits the training batch size. Complete scores, linear-probe results, and all batch sizes are in section F.

4.7 Performance under a memory budget

At the same batch size, OOB methods use the same GPU memory as their in-batch counterparts. Their advantage is that they perform well with much smaller batches, reducing the memory needed to obtain useful representations. We compare the best classification performance achievable within each GPU-memory budget on the datasets in section 4.1. Figure 7 shows that OOB methods can match or exceed the performance of their in-batch counterparts while using substantially less GPU memory within the evaluated range, making strong representations accessible under tighter memory budgets. Training times and detailed resource measurements are provided in section G.

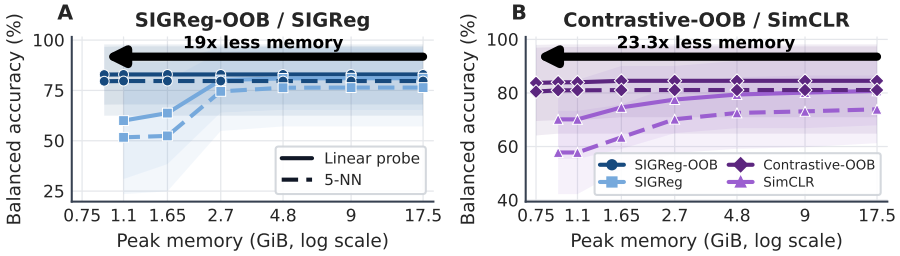

Figure 7: Best classification performance under GPU-memory budgets.

5 Conclusion and Future Work

We introduced out-of-batch (OOB) regularization, which decouples population-level representation regularization from minibatch optimization through periodically constructed dataset-level targets. Across contrastive and non-contrastive objectives, OOB more faithfully recovers the full-dataset regularization signal and improves representation quality, particularly in cases where small mini-batches poorly approximate the full population. Our results show that the active minibatch does not need to define the population over which representation geometry is regularized.

OOB replaces minibatch sampling error with a different trade-off: dataset-level state must be stored and periodically refreshed as the encoder evolves. Understanding this trade-off at larger scales remains an important direction. More broadly, our results motivate separating globally coupled representation objectives from the minibatch updates used to train the encoder, with regularizer-specific constructions providing a possible route to population-aware learning beyond the objectives studied here.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, assist in searching for relevant literature, and help write code. We did not use generative AI to generate research ideas or propose experiments. We reviewed the AI-assisted text and references, and independently inspected and executed the AI-assisted code to verify its correctness. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We describe the proposed algorithms in the main text and provide detailed architectures, training configurations, augmentations, hyperparameters, and evaluation protocols in the appendix to support independent reproduction of our results. We also document the hyperparameter-selection procedure and computational costs. We plan to release the code to further facilitate reproduction and comparison with our methods.

REFERENCES

Konstantinos Almpanakis and Anna Kreshuk. Icone: Batch independent collapse prevention for self-supervised representation learning. In *European Conference on Computer Vision*, pp. 445–463. Springer, 2026.

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.

Mahmoud Assran, Randall Balestriero, Quentin Duval, Florian Bordes, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, and Nicolas Ballas. The hidden uniform cluster prior in self-supervised learning. *arXiv preprint arXiv:2210.07277*, 2022.

Randall Balestriero. Unsupervised learning on a diet: Datum index as target free of self-supervision, reconstruction, projector head. *arXiv preprint arXiv:2302.10260*, 2023.

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

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Piotr Bojanowski and Armand Joulin. Unsupervised learning by predicting noise. In *International conference on machine learning*, pp. 517–526. PMLR, 2017.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European conference on computer vision*, pp. 139–156. Springer, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9630–9640. IEEE, 2021.

Changyou Chen, Jianyi Zhang, Yi Xu, Liqun Chen, Jiali Duan, Yiran Chen, Son Tran, Belinda Zeng, and Trishul Chilimbi. Why do we need large batchsizes in contrastive learning? a gradient-bias perspective. *Advances in Neural Information Processing Systems*, 35:33860–33875, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020a.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 15745–15753. IEEE, 2021.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3606–3613, 2014.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9568–9577. IEEE, 2021.

Peyré Gabriel and Cuturi Marco. Computational optimal transport with applications to data sciences. *Foundations and trends® in machine learning*, 11(5-6):355–607, 2019.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pp. 316–321, 2021.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann Lecun. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank. In *International conference on machine learning*, pp. 10929–10974. PMLR, 2023.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 9726–9735. IEEE, 2020.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. *arXiv preprint arXiv:2110.09348*, 2021.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Vebjorn Ljosa, Katherine L Sokolnicki, and Anne E Carpenter. Annotated high-throughput microscopy image sets for validation. *Nature methods*, 9(7):637, 2012.

Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018.

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

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Colorado J Reed, Xiangyu Yue, Ani Nrusimha, Sayna Ebrahimi, Vivek Vijaykumar, Richard Mao, Bo Li, Shanghang Zhang, Devin Guillory, Sean Metzger, et al. Self-supervised pretraining improves self-supervised pretraining. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1050–1060. IEEE, 2022.

Bryan Rodas, Natalie Montesino, Jakob Ambsdorf, David Klindt, and Randall Balestriero. Dietcp: Lightweight and data efficient self supervised continued pretraining. *arXiv preprint arXiv:2509.06990*, 2025.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pp. 606–610. IEEE, 2007.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Yucheng Tang, Dong Yang, Wenqi Li, Holger R Roth, Bennett Landman, Daguang Xu, Vishwesh Nath, and Ali Hatamizadeh. Self-supervised pre-training of swin transformers for 3d medical image analysis. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20698–20708. IEEE, 2022.

Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625*, 2019.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pp. 9929–9939. PMLR, 2020.

Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6387–6396. IEEE, 2020.

Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3733–3742, 2018.

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific data*, 10(1):41, 2023.

Chung-Yiu Yau, Hoi-To Wai, Parameswaran Raman, Soumajyoti Sarkar, and Mingyi Hong. Emc ̂2: Efficient mcmc negative sampling for contrastive learning with global convergence. *arXiv preprint arXiv:2404.10575*, 2024.

Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *European conference on computer vision*, pp. 668–684. Springer, 2022.

Zhuoning Yuan, Yuexin Wu, Zi-Hao Qiu, Xianzhi Du, Lijun Zhang, Denny Zhou, and Tianbao Yang. Provable stochastic optimization for global contrastive learning: Small batch does not harm performance. In *International Conference on Machine Learning*, pp. 25760–25782. PMLR, 2022.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021.

A REPRODUCIBILITY DETAILS

This section reports the data, optimization, augmentation, and evaluation settings used in the main experiments. Classification and retrieval results are aggregated over independent training seeds. Within each matched OOB/in-batch classification pair, we preserve the data split, augmentations, number of views, encoder, projector, optimizer, schedule, and downstream evaluator.

A.1 DATASETS AND EVALUATION PROTOCOLS

Classification. The datasets used in each collection are summarized in table 4.

Table 4: Datasets used in the classification and oracle experiments.

Collection	Domains
Natural images	CIFAR-10, CIFAR-100, DTD, ImageNet-100, Imagenette, Imagewoof, STL-10
2D Biomedical	BloodMNIST, BreastMNIST, DermaMNIST, OCTMNIST, OrganAMNIST, OrganCMNIST, OrganSMNIST, PathMNIST, PneumoniaMNIST, TissueMNIST, BBBC010, plankton(private)
3D Biomedical	AdrenalMNIST, FractureMNIST, NoduleMNIST, OrganMNIST, SynapseMNIST, VesselMNIST

For frozen evaluation, we extract deterministic, ℓ_2-normalized backbone features and use a fixed, shuffled 70/30 train/evaluation split using one seed. We report balanced accuracy from cosine k-NN ($k \in \{1, 5, 20\}$) and from a multinomial logistic-regression probe (linear probe). The latter is fit after standardizing each feature coordinate, with a maximum of 1000 iterations. Labels are never used during self-supervised pretraining.

Instance retrieval. The retrieval study uses COIL-100 (7200 images of 100 objects), UKBench (10,200 images of 2550 objects), and INRIA Holidays (1491 images grouped into 500 instances). Retrieval uses cosine similarity between normalized frozen features, with the query itself removed from the gallery. For a query q with relevant gallery set G_q, average precision is

$$\mathrm{AP}(q) = \frac{1}{|G_q|} \sum_{k=1}^{|\mathcal{G}|} P_q(k)\, \mathbf{1}[\pi_q(k) \in G_q]\,, \tag{15}$$

where $P_q(k)$ is precision at rank k and $\pi_q(k)$ is the item at that rank. mAP averages AP over queries; Recall@K is the fraction of queries with at least one relevant item among the first K results. We run two different protocols for our experiments. The transductive protocol pretrains and retrieves on the same unlabeled collection. The held-out protocol separates instance identities with a fixed 80/20 split using one seed. All retrieval methods use the same 2D encoder, image transformations, optimizer, and evaluator as the classification experiments, with four augmented views per instance for every method.

A.2 ARCHITECTURES, OPTIMIZATION, AND AUGMENTATION

The 2D experiments use a ViT-Small/16 encoder with width 384, 12 transformer blocks, and six attention heads. Inputs are 224×224. The 3D experiments use a 3D ResNet-18 and 64^3 inputs. All methods train for 100 epochs with AdamW, learning rate 10^{-4}, weight decay 0.05, gradient clipping at norm 1, and a per-step cosine schedule ending at 10^{-6}. We use the PyTorch AdamW defaults $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 10^{-8}$. From-scratch training uses fp32, with each experiment run on a single NVIDIA L40S GPU.

A.3 METHOD-SPECIFIC HYPERPARAMETERS AND COST

Both OOB methods use a fixed refresh period of 0.5 epoch. Refresh inference is batched independently of online training, with batch size 64 in 2D and 16 in 3D. SIGReg-OOB requires one deterministic dataset pass followed by $O(MN \log N)$ CPU sorting. Its persistent fp32 cache contains NM scalars. CONTRASTIVE-OOB performs 20 stochastic anchor updates, each using all pairs

Table 5: Augmentation pipelines used for self-supervised training.

Modality	Spatial transforms	Appearance and normalization
2D	Random resized crop to 224 (scale 0.2–1.0); horizontal flip	Color jitter $(0.4, 0.4, 0.4, 0.1)$, $p = 0.8$; grayscale, $p = 0.2$; Gaussian blur (kernel 23, $\sigma \in [0.1, 2.0]$); ImageNet normalization
3D	Random resized crop to 64^3 (scale 0.5–1.0); independent axis flips, $p = 0.5$; random 90° rotation in one plane, $p = 0.5$	Intensity shift in $[-0.1, 0.1]$; contrast in $[0.8, 1.2]$; Gaussian noise ($\sigma = 0.1$), $p = 0.3$; 3^3 average-pool blur, $p = 0.3$

Table 6: Method-specific settings used in the principal experiments.

Method	Configuration
SIGReg-OOB	Three-layer MLP projector with 2048-dimensional hidden and output layers, LayerNorm and ReLU after each non-final layer; $M = 256$ projections; $(\lambda_{\text{align}}, \lambda_{\text{ac}}) = (5, 1)$; projection directions resampled at every refresh.
SIGREG	The same 2048-dimensional projector
CONTRASTIVE-OOB	Two-layer ReLU projector with one backbone-width hidden layer and 128-dimensional output anchors; $(\lambda_{\text{align}}, \lambda_{\text{ac}}) = (1, 1)$; 20 anchor updates with Adam at learning rate 0.05; sampled anchor subset 256; $\tau_u = 2$, $\lambda_f = 0.1$, and previous-cache mixture weight 0.5.
SimCLR	The same two-layer 128-dimensional projector, normalized output, four augmented views in the classification experiments, and temperature 0.5.
MoCoV2	Two-layer 128-dimensional query/key projectors, four augmented views, queue size 4096, encoder momentum 0.999, and temperature 0.2.

within a sampled subset of at most 256 anchors, while its persistent cache contains $128N$ scalars. The caches live in CPU memory, and only the rows indexed by the active minibatch are transferred to the GPU. At $N = 13{,}000$, these caches occupy 12.7 MiB and 6.3 MiB, respectively. Cache construction adds wall time but does not increase the online training batch's activation memory. Details about training setup of baselines and our methods are reported in Table 6, while the augmentations we use are described in Table 5.

A.4 ABLATIONS OF THE SIGREG-OOB CONFIGURATION

We ablate the components of SIGReg-OOB on seven 2D natural/scientific domains and batches 1–64. The study comprises 1,792 evaluations from 39 configurations over 48 dataset–batch matched experiments, covering projection resampling, refresh period, the alignment and anti-collapse weights, and the number of projections. Since the sweep is not a full Cartesian product, we form controlled slices in which one factor changes while the remaining settings are held fixed. Within each slice, we retain only dataset–batch experiments shared by every variant.

The resampling, refresh-period, and alignment-weight ablations use the common configuration $p = 0.5$, $\lambda_{\text{align}} = 5$, $\lambda_{\text{ac}} = 1$, and $M = 256$, omitting the factor being varied. The anti-collapse-weight slice fixes $(p, \lambda_{\text{align}}, M) = (2, 25, 512)$, and the projection-count slice fixes $(p, \lambda_{\text{align}}, \lambda_{\text{ac}}) = (2, 25, 5)$; projection resampling is enabled in both.

Figure 8 shows mean downstream linear-probe balanced accuracy. Projection resampling produces the largest improvement. A refresh period of 0.5 epoch performs best, while one epoch remains close and overly frequent refreshes are not beneficial. The intermediate alignment weight is preferable to either extreme, and increasing the anti-collapse weight or the number of projections does not improve the matched slices. These results indicate that maintaining diverse, periodically updated targets and balancing the two loss terms matter more than increasing the raw number of projections.

B ORACLE-GRADIENT PROTOCOL

The oracle study differentiates the projected-Gaussian anti-collapse loss with respect to each probe representation. Consequently, every "gradient" is a vector in representation space, not the scalar

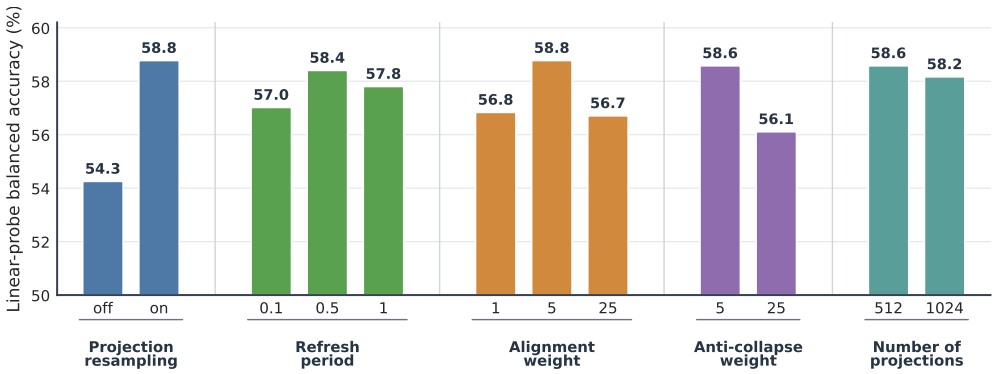

Figure 8: Controlled ablations of SIGReg-OOB. Each factor varies within a matched slice, and bars report mean linear-probe balanced accuracy over the dataset–batch cells shared by all values in that group.

loss and not the full network parameter gradient. The full-dataset reference recomputes ranks and Gaussian quantile assignments from the current representations of every training sample. The mini-batch estimator computes the same construction from a sampled batch, whereas the cached estimator reuses full-dataset assignments made at an earlier encoder state. At every cache age, the reference itself is recomputed at the current encoder state; cached agreement is therefore measured against the current population objective rather than against the reference at refresh.

The canonical experiment uses all 2D and 3D datasets considered in this study. We evaluate checkpoints at epochs 0, 50, and 100 over three seeds. Each condition uses 16 probe samples, 16 sampled contexts, and 256 projection directions. Minibatch sizes are 1–64 and cache ages are $s \in \{0, 1, 2, 4, 8, 16, 32, 64, 128\}$ optimizer updates. Direction is measured by gradient cosine similarity, and scale-sensitive disagreement by relative root-mean-square error.

At $B = 1$, mean cosine agreement at epochs 0, 50, and 100 is respectively -0.047, -0.417, and -0.362; at $B = 64$ it rises to 0.972, 0.903, and 0.939. A fresh cache agrees exactly by construction. At age 128, mean cached agreement at the default learning rate is 0.453, 0.720, and 0.912 across the three checkpoints. At the matched low-rate control (10^{-6}), the values are 0.692, 0.909, and 0.906, with the largest improvements at epochs 0 and 50. Cache lifetime is therefore controlled by how rapidly the encoder is updated, not by update count alone.

Figure 9 verifies that the ordering between limited batch context and fresh dataset-level context is not an artifact of a single dataset size. The oracle remains a reference for estimating this chosen finite-population regularizer.

C Detailed frozen-classification results

Figure 10 summarizes the controlled 2D MedMNIST size sweep. The advantage of OOB regularization is largest at small batches and persists as the number of training examples grows. Tables 7 and 8 report the per-dataset results after averaging over a common set of batch sizes. The 2D biomedical and 3D tables average $B \in \{2, 4, 8, 16, 32, 64\}$, while the natural-image table averages $B \in \{2, 4, 8, 16, 32\}$. The full ImageNet-100 row instead averages its available batches 64 and 128. Each entry gives 5-NN and linear-probe balanced accuracy, in that order.

Across the controlled size sweep, the OOB variants retain their strongest advantages in the low-batch regime rather than relying on a particular number of training examples. The per-dataset results below also show that these gains are distributed across domains rather than being driven by a single task.

The natural-image and 3D results in table 8 broaden the comparison beyond the controlled 2D size sweep while retaining the same frozen-evaluation protocol.

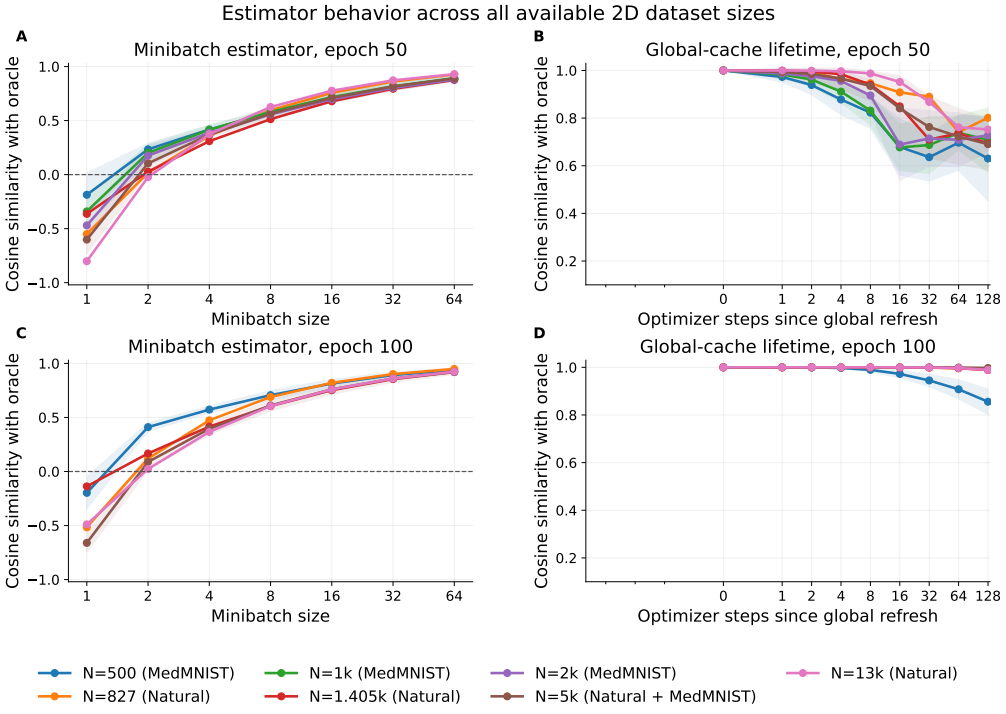

Figure 9: Robustness of the oracle comparison to dataset size. The MedMNIST sweep uses 500, 1k, 2k, and 5k training examples. Panels A and C evaluate the minibatch estimator, while B and D follow global-cache lifetime at epochs 50 and 100, respectively.

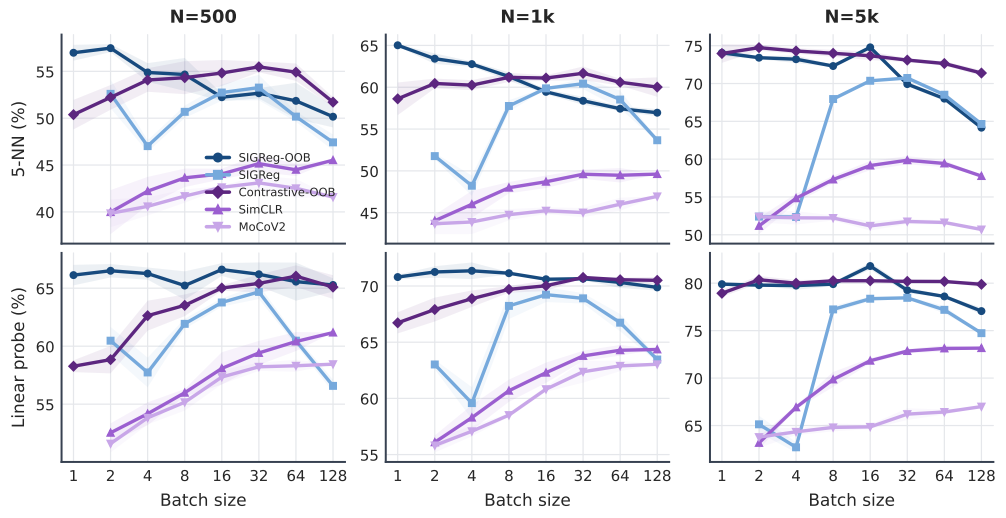

Figure 10: Frozen classification across MedMNIST training-set sizes. Shading denotes one standard deviation across training seeds.

Table 7: Per-dataset 2D MedMNIST balanced accuracy (%) across training-set sizes, averaged over batch sizes. Entries are linear probe results. Within each method family, the best result is bold; the best result overall is underlined.

(a) $N = 500$

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive-OOB	VICReg	IConE	IConE-SIGReg	BYOL	SwAV	DINO
BloodMNIST	82.4	**84.4**	68.9	68.1	**78.2**	76.3	81.1	80.8	72.9	75.7	72.6
BreastMNIST	71.3	**72.5**	66.6	65.2	**72.0**	70.9	68.8	68.5	67.8	68.1	65.5
DermaMNIST	54.7	**60.0**	50.9	48.6	**59.5**	52.4	45.2	45.2	49.1	49.3	46.3
OCTMNIST	48.2	**52.6**	41.2	39.3	**41.8**	48.9	42.9	44.2	41.5	43.8	38.2
OrganAMNIST	61.0	**70.7**	54.3	52.8	**66.9**	61.0	73.1	72.6	55.2	56.8	52.5
OrganCMNIST	58.9	**65.6**	51.2	47.8	**63.5**	64.3	70.9	71.5	50.2	51.6	53.0
OrganSMNIST	55.3	**56.4**	46.8	44.4	**56.0**	52.9	55.2	55.8	47.2	48.0	46.6
PathMNIST	70.4	**76.5**	68.0	70.9	**74.4**	74.0	66.0	65.5	68.2	72.1	64.2
PneumoniaMNIST	86.7	**88.7**	85.4	84.2	**88.3**	88.0	85.1	85.4	87.1	86.2	82.6
TissueMNIST	28.4	**34.1**	27.1	**31.1**	30.3	30.0	27.9	28.7	29.9	30.9	28.5

(b) $N = 1\mathbf{k}$

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive-OOB	VICReg	IConE	IConE-SIGReg	BYOL	SwAV	DINO
BloodMNIST	86.5	**91.7**	79.6	79.9	**90.6**	84.1	84.2	84.9	81.7	82.0	77.6
BreastMNIST	72.4	**74.6**	69.9	67.5	**71.5**	71.2	72.7	71.3	71.0	70.6	66.5
DermaMNIST	52.2	**58.0**	50.2	48.4	**57.7**	51.9	48.3	48.4	48.8	48.1	46.3
OCTMNIST	53.6	**59.5**	45.4	45.4	**51.7**	53.4	45.5	44.8	44.6	48.9	41.5
OrganAMNIST	72.1	**77.2**	61.4	61.1	**76.4**	67.4	77.4	76.9	59.6	61.7	53.0
OrganCMNIST	70.4	**76.5**	60.4	57.3	**76.8**	71.3	80.1	79.2	59.4	61.1	57.0
OrganSMNIST	59.5	**60.8**	49.2	48.9	**61.8**	54.5	59.4	59.5	48.8	49.2	47.4
PathMNIST	67.7	**79.0**	67.6	67.1	**76.9**	69.1	56.2	55.2	66.2	64.5	60.1
PneumoniaMNIST	88.0	**91.4**	84.1	81.3	**89.5**	85.4	86.8	85.7	84.9	84.4	80.8
TissueMNIST	35.6	**41.3**	34.4	32.4	**41.7**	34.9	31.2	33.3	33.4	32.8	30.8

(c) $N = 2\mathbf{k}$

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive-OOB	VICReg	IConE	IConE-SIGReg	BYOL	SwAV	DINO
BloodMNIST	90.1	**95.4**	**82.1**	80.2	81.9	85.9	88.3	88.0	83.7	83.4	77.2
BreastMNIST	73.6	**75.6**	70.8	68.2	**73.6**	73.4	73.2	72.1	71.0	72.2	65.1
DermaMNIST	53.5	**59.9**	52.5	48.4	**52.8**	53.8	49.0	48.3	47.8	45.2	43.3
OCTMNIST	58.1	**66.0**	**46.4**	44.6	46.1	56.1	48.0	48.4	46.5	48.1	41.4
OrganAMNIST	75.0	**81.4**	**66.8**	61.8	66.3	69.9	80.3	79.6	61.1	63.2	57.2
OrganCMNIST	76.2	**82.6**	**69.2**	61.5	65.8	73.9	80.3	80.3	64.7	62.6	58.9
OrganSMNIST	62.6	**69.3**	58.2	53.5	**60.6**	60.6	62.9	62.2	55.2	53.7	49.7
PathMNIST	75.5	**85.3**	**73.2**	69.1	72.2	73.1	67.4	67.8	68.3	68.1	60.1
PneumoniaMNIST	91.7	**94.2**	**89.3**	87.3	86.8	90.1	90.6	89.5	89.1	88.0	84.1
TissueMNIST	37.1	**46.5**	**34.2**	32.5	33.1	38.0	30.1	30.4	32.2	31.8	27.4

(d) $N = 5\mathbf{k}$

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive-OOB	VICReg	IConE	IConE-SIGReg	BYOL	SwAV	DINO
BloodMNIST	93.2	**97.5**	90.3	87.7	**97.7**	88.4	91.3	92.3	87.7	90.0	80.9
BreastMNIST	73.3	**75.4**	71.0	67.7	**73.8**	72.8	73.1	73.8	72.6	71.5	66.1
DermaMNIST	52.9	**61.5**	50.5	45.9	**62.3**	50.6	47.6	46.1	45.5	44.3	40.4
OCTMNIST	63.5	**76.5**	56.7	51.5	**77.1**	60.7	51.5	51.7	55.2	52.1	43.2
OrganAMNIST	81.8	**88.2**	75.7	69.8	**89.1**	76.1	80.7	82.3	67.0	64.4	56.6
OrganCMNIST	78.9	**85.0**	76.5	68.4	**86.1**	74.9	79.8	80.7	69.6	65.2	59.5
OrganSMNIST	68.4	**75.0**	66.0	61.3	**76.6**	64.4	64.6	68.3	58.1	58.5	51.0
PathMNIST	79.8	**92.2**	76.7	73.8	**92.9**	74.6	73.3	72.5	70.9	72.3	62.5
PneumoniaMNIST	92.4	**95.8**	89.5	87.2	**96.0**	89.1	89.2	87.6	89.5	87.3	82.0
TissueMNIST	39.6	**49.7**	36.3	34.8	**50.5**	38.0	32.8	32.3	33.9	32.6	28.6

Table 8: Per-dataset balanced accuracy (%) on natural/scientific 2D and 3D MedMNIST datasets, averaged over batch sizes. Entries are 5-NN / linear probe. Within each method family, the best result is bold; the best result overall is underlined.

(a) Natural/scientific 2D

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive -OOB
BBBC010	**90.9** / **93.8**	90.4 / 91.5	89.9 / 90.9	**90.8** / 90.9	90.0 / **91.2**
CIFAR-10	49.2 / 60.0	**79.7** / **82.2**	**76.2** / **81.1**	68.2 / 73.8	74.9 / 79.4
CIFAR-100	20.7 / 33.5	**42.2** / **53.9**	**39.1** / **53.1**	34.6 / 46.9	33.4 / 49.4
DTD	17.5 / 25.2	**26.0** / **32.7**	26.5 / 35.8	22.3 / 29.1	**32.4** / **37.0**
ImageNet-100 (full)	44.3 / 53.9	**52.4** / **60.6**	– / –	– / –	– / –
ImageNet-100-5k	39.3 / 54.4	**55.9** / **67.8**	53.8 / 65.3	49.2 / 63.2	**63.9** / **69.5**
Imagenette	39.5 / 51.9	**58.2** / **66.6**	56.5 / 65.9	45.8 / 58.4	**66.0** / **68.4**
Imagewoof	20.1 / 30.3	**28.2** / **39.6**	28.2 / 40.3	22.9 / 34.5	**34.5** / **42.1**
Plankton	41.3 / 42.8	**42.6** / **45.8**	**43.1** / **45.6**	40.5 / 42.6	42.9 / 45.5
STL-10	43.4 / 52.4	**61.5** / **67.5**	59.9 / 66.4	50.3 / 59.6	**64.6** / **69.9**

(b) 3D MedMNIST

Dataset	SIGReg	SIGReg-OOB	SimCLR	MoCoV2	Contrastive -OOB
AdrenalMNIST3D	54.4 / 57.6	**57.1** / **64.7**	55.6 / 60.0	56.5 / 62.9	**58.8** / **64.8**
FractureMNIST3D	**37.7** / 40.5	36.5 / **41.3**	**37.8** / **42.1**	37.6 / 41.4	36.6 / 39.2
NoduleMNIST3D	64.7 / 66.5	**72.6** / **76.7**	65.1 / 71.6	68.2 / 75.7	**72.2** / **76.4**
OrganMNIST3D	44.8 / 63.4	**56.5** / **83.1**	42.2 / 60.4	55.8 / **80.3**	**57.5** / 75.8
SynapseMNIST3D	55.5 / 56.8	**57.6** / **59.0**	**57.6** / 58.8	54.0 / **60.4**	54.1 / 58.9
VesselMNIST3D	51.2 / 50.7	**53.4** / **58.9**	52.3 / 52.2	**54.7** / **56.0**	53.6 / 55.5

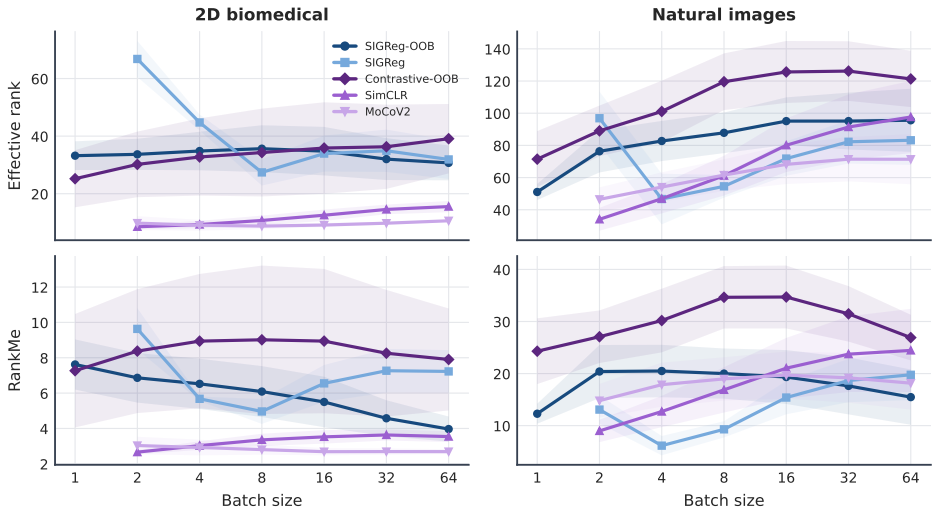

Figure 11: Effective rank and RankMe of frozen representations. Lines average over the common datasets and bands show standard errors across datasets.

C.1 Representation-rank diagnostics

Figure 11 provides a useful qualification. Contrastive-OOB has much higher rank than SimCLR and MoCoV2 in the smallest-batch regime, consistent with stronger prevention of dimensional collapse. The Gaussian pair is more mixed: SIGReg-OOB can have lower rank than SIGReg on 2D biomedical data while matching or exceeding its downstream accuracy, and is comparable or higher on natural images. Rank therefore diagnoses one aspect of collapse but does not by itself explain semantic performance.

D DETAILED CLASS-IMBALANCE RESULTS

We generate controlled imbalance by retaining a $1/k$ fraction of the samples from designated classes, for $k \in \{2, 5, 8, 10, 15, 20, 25\}$. The SSL algorithms remain label-free. Class labels are used only after pretraining to report macro recall over the downsampled classes, worst-class recall, and the minority–majority recall gap. Figure 12 reports the 2D results at batch size 64, while fig. 13 condenses the more stringent tail metrics for the contrastive methods.

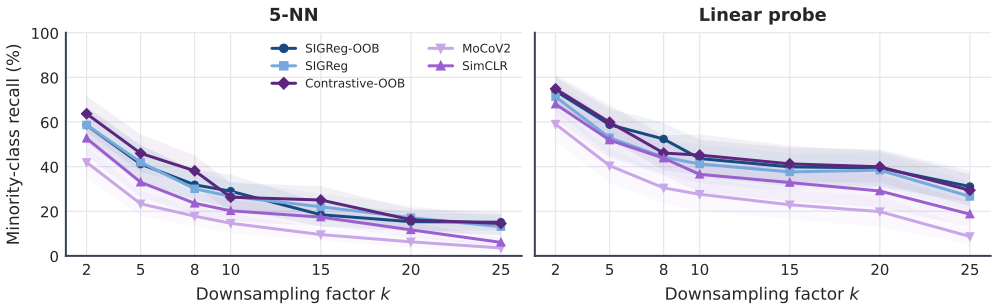

Figure 12: Minority-class recall for the controlled-imbalance 2D experiments at batch size 64: 5-NN (left) and linear probe (right).

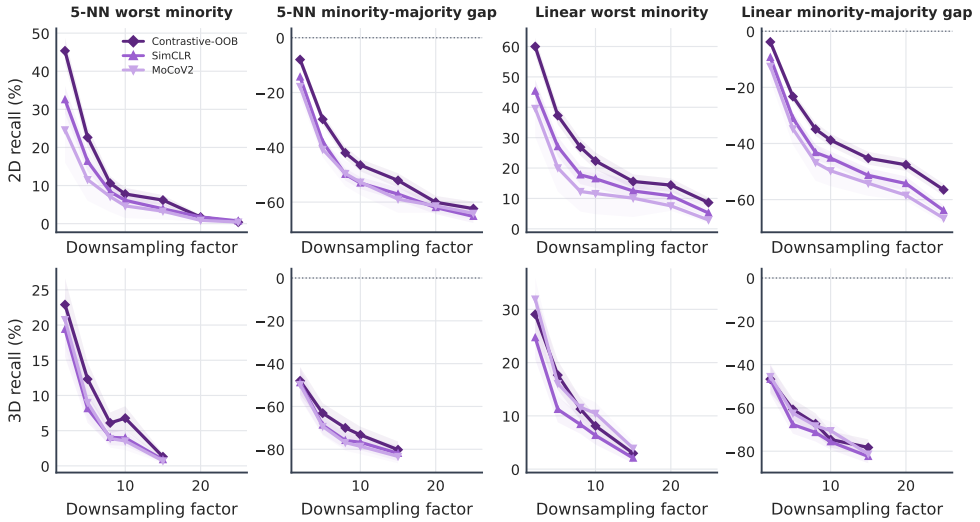

Figure 13: Worst-class recall and minority–majority recall gap for Contrastive-OOB, SimCLR, and MoCoV2. Lines average dataset–batch experiments and bands show standard errors across those results.

The OOB gains are not restricted to average classification accuracy: they are visible for the underrepresented classes and, in many settings, for the worst class. Nevertheless, an empirical cache preserves the empirical population's frequency weighting. The label-aware oracle diagnostic in the main paper therefore separates two claims: OOB state can estimate a chosen population objective more faithfully, but changing that objective to be class-balanced requires an explicit weighting decision.

E INSTANCE-RETRIEVAL RESULTS

The held-out evaluation confirms that the transductive improvements are not solely memorization of the training instances (Table 9 and fig. 14). CONTRASTIVE-OOB gives the highest average

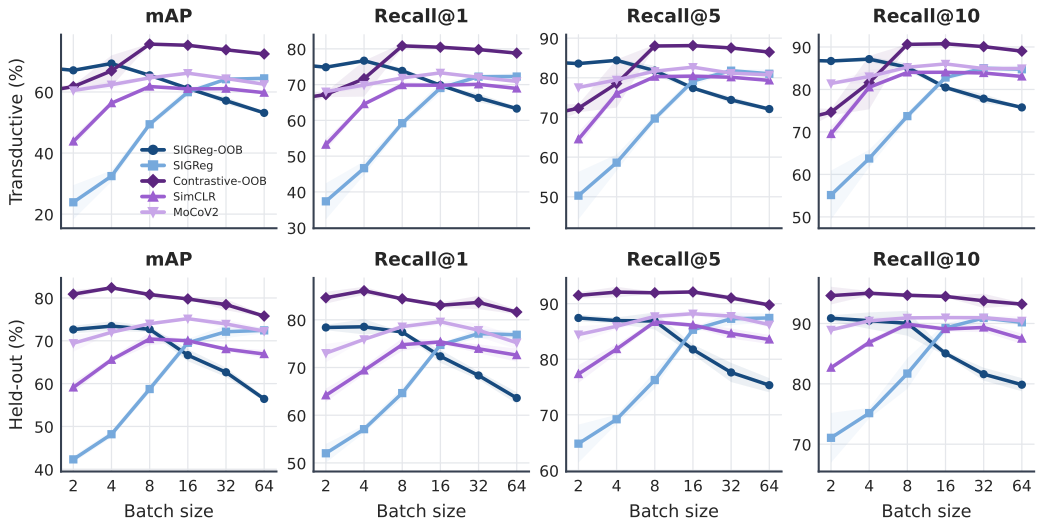

Figure 14: Transductive and held-out retrieval for mAP and Recall@$1/5/10$. For each seed, scores are first averaged over the three datasets; lines then show the three-seed mean and bands show one standard deviation.

Table 9: Retrieval performance (%), averaged over the three datasets, batches 2–64, and three training seeds within each protocol. Bold denotes an OOB result that exceeds its corresponding in-batch comparator: SIGReg-OOB versus SIGReg, or Contrastive-OOB versus both SimCLR and MoCoV2.

Protocol	Metric	SIGReg	SIGReg-OOB	SimCLR	Contrastive-OOB	MoCoV2
Transductive	mAP	49.1	**62.2**	57.3	**71.0**	63.4
	R@1	59.5	**70.8**	66.1	**76.5**	71.0
	R@5	70.1	**79.0**	76.8	**83.5**	80.6
	R@10	74.2	**82.2**	80.8	**86.1**	84.3
Held-out	mAP	60.6	**67.4**	66.7	**79.7**	72.8
	R@1	67.1	**73.1**	71.7	**83.9**	76.7
	R@5	78.4	**82.7**	83.4	**91.4**	86.7
	R@10	83.0	**86.3**	87.6	**94.3**	90.4

mAP and recall in both protocols, while SIGReg-OOB improves over SIGReg on every aggregate retrieval metric.

F Continued pretraining with DINOv2

Continued-pretraining runs start from public DINOv2 ViT-S/B/L/G checkpoints and use two augmented views for 100 epochs. The backbone and projector learning rates are 10^{-5} and 10^{-4}, respectively; weight decay is 0.05 and the gradient norm is clipped at 1. OOB refreshes occur every 0.5 epoch. Refresh batch sizes for ViT-S/B/L/G are 128, 64, 16, and 4 to accommodate their different activation footprints. Experiments use an NVIDIA A100 GPU.

The per-dataset results (Figure 15) support the main conclusion: at small batches, OOB regularization is more likely to retain or improve the pretrained representation than its in-batch counterpart. The benefit nevertheless depends on the method, encoder, and target dataset, consistent with the central claim being about access to population context rather than guaranteed improvement for every pretrained representation.

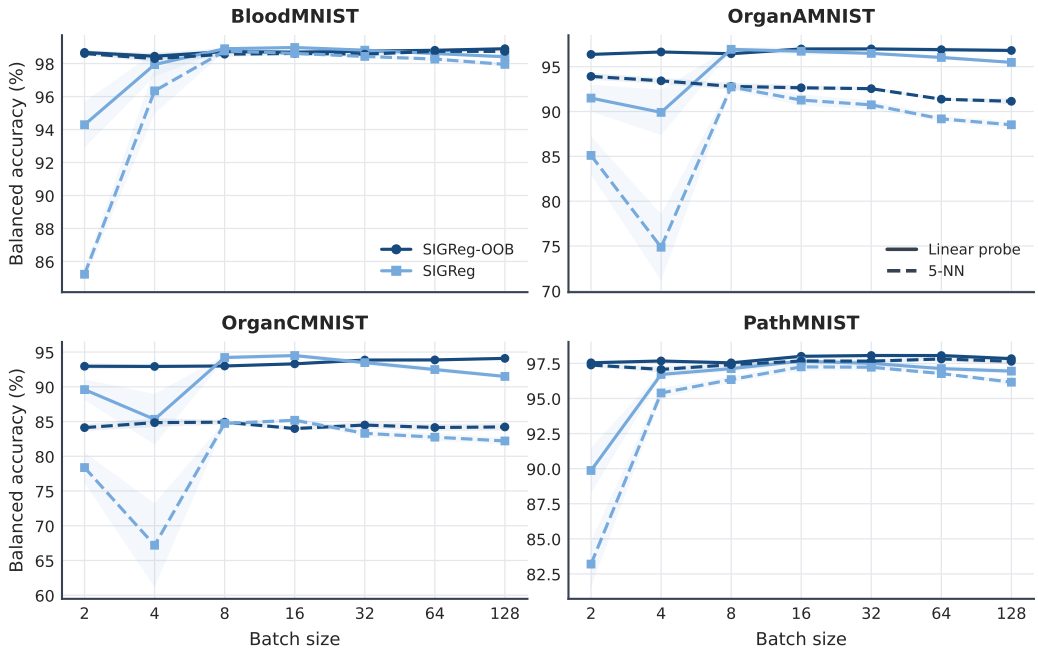

Figure 15: Per-dataset DINOv2 continued-pretraining performance on biomedical datasets for SIGReg and SIGReg-OOB. Solid lines show linear-probe accuracy and dashed lines show 5-NN accuracy. Each point averages DINOv2-S/B/L/G encoders; bands show standard errors across encoder sizes.

G Computational measurements

We benchmark each method on one NVIDIA L40S GPU for 100 epochs (Figure 16). The five-method benchmark spans BloodMNIST at 500, 1k, 2k, and 5k training examples and STL-10. We evaluate batches 1–64 where each method is defined. Peak allocated GPU memory is measured during training; wall time includes refresh passes and anchor construction.

At a matched batch size, SIGReg-OOB and Contrastive-OOB have essentially the same peak allocated GPU memory as their direct SIGReg and SimCLR counterparts. They add approximately 10% and 4% wall time, respectively, due to periodic cache construction.

H Additional analyses

H.1 Convergence over pretraining time

We record frozen 5-NN and linear-probe performance after every epoch for BloodMNIST-5k, STL-10, and ImageNet-100-5k. Figure 17 and table 10 shows the difference in improvement between OOB and in-batch variants as optimization proceeds. The Gaussian OOB advantage grows throughout training and the contrastive advantage emerges early and remains positive on average through epoch 200.

Table 10: Mean OOB gain in percentage points at selected epochs.

Comparison	Probe	Epoch 25	Epoch 50	Epoch 100	Epoch 200
SIGReg-OOB − SIGReg	5-NN	+9.09	+11.94	+13.88	+14.43
SIGReg-OOB − SIGReg	Linear	+6.91	+9.14	+10.80	+11.62
Contrastive-OOB − SimCLR	5-NN	+1.58	+3.68	+3.71	+2.43
Contrastive-OOB − SimCLR	Linear	+0.89	+1.73	+1.71	+0.47

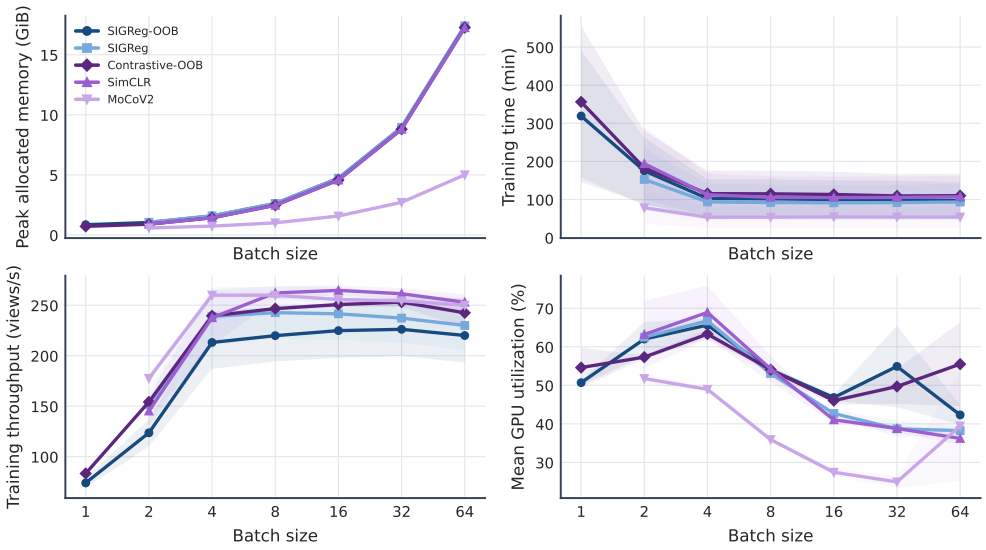

Figure 16: Detailed resource measurements across dataset sizes and batches. Lines average the datasets evaluated for each method, and bands show standard errors across datasets.

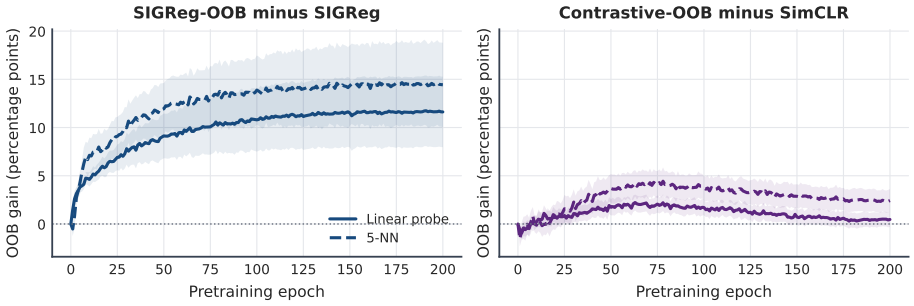

Figure 17: OOB improvement over the corresponding in-batch method during pretraining. Solid lines show linear-probe accuracy and dashed lines show 5-NN accuracy. Lines average matched dataset–batch cells and bands show standard errors across those cells.