DISENTANGLING REGULARIZATION EFFECTS IN SELF-SUPERVISED LEARNING THROUGH CONTROLLED GRADIENT INTERVENTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-objective self-supervised visual learning couples auxiliary losses and geometric regularizers through shared encoders, yet regularization gains need not translate into gains over a base-loss reference. Existing studies document configuration dependence, but removing an auxiliary loss changes updates to several parameter groups, obscuring the contribution of direct encoder updates. We therefore study how direct auxiliary encoder updates influence regularization effects. At the same computational state, we retain or remove only the auxiliary encoder gradient while matching forward losses and all other parameter gradients. Crossing this intervention with regularizer inclusion, alongside a base-loss reference, distinguishes regularization effects, combination effects, and update-path dependence. Paired experiments cover controlled-background predictive learning and natural-image self-distillation across regularizers, auxiliary weights, and training stages. Results show that removing auxiliary encoder gradients reverses the mean effect of Gram regularization on accuracy from +1.98 to -1.70 percentage points (pp) in five-fold linear evaluation of Caltech–UCSD Birds-200-2011 pretraining images after 20 epochs, consistently across six runs. This reversal also occurs with covariance regularization and with Gram at a lower auxiliary weight; these combinations remain below the base-loss reference. On Tiny ImageNet after 100 epochs, Kozachenko–Leonenko regularization lowers mean held-out accuracy by 0.50 pp with auxiliary encoder updates retained, although the full DINOv2-style combination exceeds the reference by 0.92 pp. On a fixed validation set, the interaction changes by -1.04 pp between epochs 20 and 100 (95% confidence interval: -1.84 to -0.25). These comparisons inform regularizer reuse and configuration selection under specified objectives, gradient paths, and training budgets.

1 INTRODUCTION

Self-supervised learning (SSL) constructs training objectives from the data itself and learns representations during pretraining without manual labels. Representative methods include contrastive learning (Chen et al., 2020), non-contrastive learning (Grill et al., 2020; Chen & He, 2021), and methods based on image joint-embedding predictive architectures (Assran et al., 2023). In these methods, design choices concerning network architectures, training objectives, and gradient update rules are interrelated, requiring component effects to be evaluated within specific configurations (Wittscher, 2026). Clarifying the effects of regularizers under different objective combinations and update paths is important for evaluating representation learning methods and selecting training configurations.

Previous studies have demonstrated this configuration dependence from different perspectives. In linear non-contrastive models, the predictor, exponential moving average updates of the target network, and weight decay jointly affect learning dynamics (Tian et al., 2021). The same prototype regularization configuration can increase or decrease accuracy across different SSL methods (Arteaga et al., 2026); the gains from class-separability and effective-dimensionality constraints also vary across SSL methods (Dai et al., 2026). Therefore, regularization gains in a single configuration are insufficient to determine the effects after the objective combination or update path changes.

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.

For training systems comprising a base loss, an auxiliary loss, and a geometric regularizer, three questions need to be distinguished: whether the regularizer improves the current configuration, whether the full combination of the auxiliary loss and regularizer outperforms a reference containing only the base loss, and whether the regularization effect depends on direct encoder updates from the auxiliary loss. These questions concern component gains, combination gains, and the dependence of component gains on update paths, respectively. When the auxiliary loss updates both the encoder and the predictor or projection head, disabling this loss removes multiple gradient paths, making it difficult to assess the role of encoder updates separately. Clarifying this dependence provides an important basis for determining the conditions under which a regularizer is applicable, evaluating component effects, and selecting training configurations.

To examine how direct encoder updates from the auxiliary loss affect regularization effects, we draw on existing approaches to gradient analysis and intervention (Tao et al., 2022; Halvagal et al., 2023) to construct matched gradient comparisons. At the same computational state, we keep forward loss values and all other parameter gradients identical, retain or remove only the direct gradient from the auxiliary loss to the encoder, and cross this intervention with the regularizer switch to compare regularization effects using linear classification accuracy. We also introduce a reference containing only the base loss to separately evaluate the regularizer's effect within its configuration and the performance of the full combination relative to the base objective, and compare how these effects vary across training configurations and stages.

This paper makes three contributions. First, we use matched gradient intervention to identify the conditional dependence of regularization effects on encoder updates from the auxiliary loss, incorporating update paths into the evaluation of regularizer applicability. Second, we propose a diagnostic design that combines a base-loss reference with a regularizer switch to distinguish regularization effects, full-combination effects, and the dependence of the former on update paths, providing a basis for component selection and the choice of combinations. Third, we analyze changes in regularization interactions across training stages and distinguish early observations from endpoint evaluation, providing experimental evidence for selecting regularizers under a target training budget.

2 Related Work

Configuration dependence of regularization and component attribution. A survey of SSL design choices reviews the dependencies among data, training objectives, and network architectures (Wittscher, 2026). Theoretical studies incorporate data augmentation, inductive bias, and regularization into analyses of representations and downstream generalization (Cabannes et al., 2023), and establish contrastive-learning error bounds that explicitly depend on augmentation properties (Cui et al., 2025). These design relationships also appear in invariant and equivariant representation learning: separating the two types of representations and introducing a transformation-conditioned hypernetwork predictor (Garrido et al., 2023), or predicting augmentation-induced embedding displacements (Devillers & Lefort, 2023), can both combine invariance objectives with equivariant learning. However, regularization gains within component combinations still depend on the specific configuration. Research on joint and decoupled optimization of prototypes and encoders has found that the same prototype regularization configuration can increase or decrease accuracy across different SSL methods (Arteaga et al., 2026). Analyses of this conditional dependence also address constraint combinations and where constraints are applied: research on dense representations analyzes structural degradation through class separability and effective dimensionality, and compares the separate and combined effects of these two types of constraints (Dai et al., 2026); research on soft equivariance regularization compares how different levels of representation in intermediate spaces affect equivariance metrics and linear classification accuracy (Lee et al., 2026). We further take direct encoder updates from the auxiliary loss as the intervention target and cross matched gradient comparisons with the regularizer switch to examine the dependence of regularization effects on this update path.

Predictors, projection heads, and matched gradient intervention. Previous studies have analyzed the effects of projection heads and representation extraction layers on downstream performance (Bordes et al., 2022; Xue et al., 2024), and revealed the role of branch asymmetry in non-contrastive learning from the perspectives of learning dynamics and feature rank (Tian et al., 2021; Zhuo et al., 2023; Ponce et al., 2026). To further separate the effects of different updates, Uni-

Grad analyzes the gradient composition of SSL by replacing or removing gradient terms (Tao et al., 2022); the Euclidean isotropic loss changes gradients of online representations while keeping the forward loss value unchanged (Halvagal et al., 2023). Control over the update process is also used to improve multi-task and auxiliary-task optimization, including projecting conflicting gradients (Yu et al., 2020), merging training branches based on validation performance (Jiang et al., 2023), and updating shared and task-specific parameters in groups (Jeong & Yoon, 2025). We further use gradient intervention for effect attribution: at the same computational state, we match forward loss values and update rules for non-encoder parameters, retain or remove only the direct gradient from the auxiliary loss to the encoder, and cross this intervention with the regularizer switch to distinguish the effect of the update itself from its influence on regularization effects.

Geometric objectives and task performance. Existing methods learn representations through cross-correlation, variance and covariance constraints, or whitening (Zbontar et al., 2021; Bardes et al., 2021; Ermolov et al., 2021). This idea has also been extended to predictive learning and self-distillation by introducing variance and covariance constraints (Mo & Tong, 2024) and coding-rate regularization (Wu et al., 2025), respectively. However, achieving geometric objectives still needs to be distinguished from task gains. Studies of dimensional collapse and whitening losses analyze the causes of representation degradation and full-rank conditions (Jing et al., 2021; Huang et al., 2024b); further research shows that high global dimensionality does not guarantee the absence of local collapse (Huang et al., 2024a), and that the effect of collapse on task performance also depends on the relationship between augmentation and the task (Ziyin et al., 2022). This geometric state also changes during training: research on stepwise learning derives the gradual growth of representation dimensionality in linear models and observes corresponding phenomena in deep-network experiments (Simon et al., 2023). Accordingly, we combine geometric measurements, linear classification evaluation under a fixed protocol, and matched gradient intervention to separately assess the attainment of geometric objectives, classification gains, and their dependence on auxiliary updates. Through paired comparisons across stages within the same training run, we distinguish stage-dependent changes in mean effects from differences across training runs.

3 Experimental Design and Evaluation Methods

Matched gradient intervention and a base-loss reference distinguish regularization effects, combination effects, and dependence on direct auxiliary encoder updates (Figure 1).

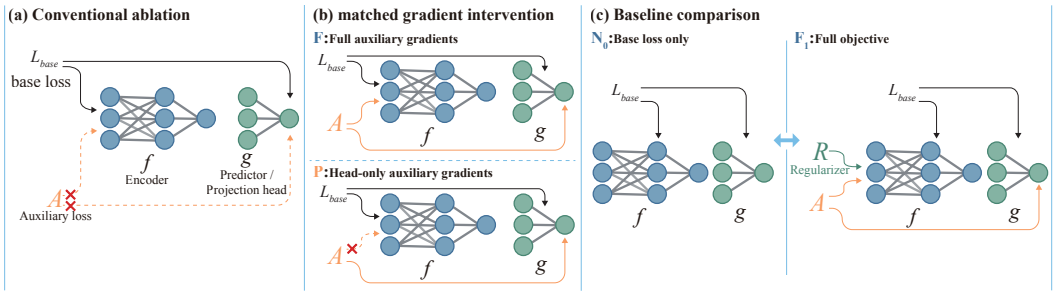


Figure 1: Matched gradient intervention and baseline comparison.

3.1 Training Objectives

The SSL objective comprises a base loss, an auxiliary loss, and a regularizer:

$$L = L_{\text{base}} + a\alpha A + \lambda_b R, \qquad a, b \in \{0, 1\}, \quad \lambda_0 = 0. \tag{1}$$

Here, L_{base}, A, and R denote the base SSL loss, auxiliary loss, and encoder-representation regularizer; a, b switch the latter two on/off. The full auxiliary weight $\alpha = 1$ (0.25 in the extension), and λ_1 is fixed within pairs. The target encoder uses stop-gradient and exponential moving average (EMA) updates.

Predictive SSL aligns online and target representations across four backgrounds sharing each image's foreground. L_{base} uses smooth L_1 (SmoothL1) token-prediction loss; A aligns state-centered residuals with target-derived whitening, combining instancewise and aggregated terms. Targets and whitening transforms use stop-gradient; both losses average two prediction directions equally (Appendix B). The random-group Gram regularizer constrains four fixed, class-independent group means:

$$R_g = \left\| \frac{H}{\operatorname{tr}(H)} - \frac{C}{3} \right\|_F^2. \tag{2}$$

where $U \in \mathbb{R}^{4\times 384}$ contains centered group means of image representations averaged over states, masks, and tokens; $H = UU^\top/384$, $C = I_4 - \mathbf{1}\mathbf{1}^\top/4$, I_4 is the identity, and $\mathbf{1}$ is the all-ones vector. The trace and Frobenius norm are denoted by tr and $\|\cdot\|_F$. Gram is averaged over both directions.

An ungrouped covariance regularizer (cov) replaces Gram, constraining variance floors and off-diagonal correlations (Eq. (7)); initialization-calibrated scale and weight remain fixed across runs (Appendix B.2). Six paired Fcov/Pcov runs yield 12 new 20-epoch trajectories, sharing Gram's F0/P0 baselines and official training/test evaluation. Effective rank is the entropy exponential of normalized covariance eigenvalues; total variance is the covariance trace.

3.2 Matched Gradient Intervention

F and P use the same auxiliary weight α, with raw gradients

$$\begin{aligned} g_\theta^{(\beta,b)} &= \nabla_\theta (L_{\text{base}} + \lambda_b R) + \beta\alpha \nabla_\theta A, \\ g_\phi^{(\beta,b)} &= \nabla_\phi (L_{\text{base}} + \alpha A), \qquad \beta = 1\ (\text{F}), \quad \beta = 0\ (\text{P}). \end{aligned} \tag{3}$$

Here, θ, ϕ denote online encoder and predictor/projection-head parameters; g_θ, g_ϕ are pre-clipping, pre-optimization gradients; ∇ is the gradient operator; and β weights direct auxiliary encoder gradients. N retains only base-loss and regularizer gradients; P also retains auxiliary updates to ϕ. Crossing F, P, N with $b = 0, 1$ gives six conditions: F/P estimate how auxiliary encoder gradients modify regularization effects, while N provides the base-loss reference for combination effects.

F/P matches update rules at fixed model/target parameters, inputs, masks, and randomness. Forward losses and ϕ gradients match; only direct auxiliary θ gradients differ. Single-step checks also fix optimizer and precision-scaler states.

We cache encoder gradients of $L_{\text{base}} + \lambda_b R$, then backpropagate the common objective. F retains full gradients; P restores the cache. Separate θ/ϕ clipping preserves matched ϕ updates; actual updates are checked at the same state. Subsequent parameters, optimizer states, and EMA trajectories evolve within each condition: F/P outcomes measure cumulative training effects, not matched trajectories.

3.3 Effect Definitions

For each independent training run, we compute the paired effects

$$\delta_X = Y_{X,1} - Y_{X,0}, \qquad D = \delta_F - \delta_P, \qquad G = Y_{F,1} - Y_{N,0}. \tag{4}$$

Here, $Y_{X,b}$ is frozen-target-encoder linear accuracy under $X \in \{\text{N}, \text{F}, \text{P}\}$; b switches regularization. Thus, δ_X measures regularization's accuracy effect under X, with all auxiliary gradients retained in F.

The interaction D measures how the auxiliary encoder gradient switch changes regularization effects; $D > 0$ indicates a larger effect in F. The reversal studied here is $\delta_F > 0$, $\delta_P < 0$: regularization raises accuracy in F and lowers it in P.

The combination effect G compares full-combination F1 with base-loss N0. Since $G = (Y_{F,0} - Y_{N,0}) + \delta_F$, its sign can differ from δ_F.

3.4 Training Budget and Auxiliary Weight

The six paired CUB F/P runs continue to 75 epochs without learning-rate resets. We vary only $\beta \in \{0, 0.5, 1\}$ in Eq. (3), fixing the predictor coefficient at 1 and both loss weights. Endpoints reuse CUB75 models, paired with the midpoint within runs.

Reducing α from 1 to 0.25 scales both encoder and predictor auxiliary updates, unlike varying only β; Gram's weight remains 1. All four F/P conditions restart from corresponding initial states: six paired runs, 24 trajectories, 20 epochs, and the same official-test evaluation.

The 37-class Oxford-IIIT Pet comparison (Parkhi et al., 2012) uses CUB's background relations and F/P controls: six independent runs of 27900 steps (229 per epoch; approximately 121.83 epochs). Linear classifiers use official trainval/test splits; other settings match CUB.

Figures 2, 3, 4, and 5 use paired run-level effects from Eq. (4); Appendix E details intervals. CUB uses official-test evaluation; Tiny ImageNet uses 100-epoch held-out endpoints and same-validation-set stage comparisons.

3.5 Data, Training Settings, and Evaluation Protocols

Controlled backgrounds vary auxiliary inputs while preserving foregrounds and labels. Given background effects on learning (Geirhos et al., 2020; Wu et al., 2026), all six conditions share background generation and prediction tasks.

CUB (Wah et al., 2011) uses four background states sharing each foreground and crop, I-JEPA's encoder–predictor framework (Assran et al., 2023), a randomly initialized ViT-S/16 Vision Transformer (ViT) (Dosovitskiy et al., 2021), and separate predictors for both directions. Enabled Gram weight is $\lambda_1 = 1$. Six conditions share images, initialization, sampling, masks, and schedules, differing only in auxiliary-gradient recipients and regularizer inclusion. The prespecified endpoint is 20 epochs; stage analysis continues the same schedule to 75.

Each condition has six paired runs. Frozen target encoders yield eight representations per image (four states, two directions), treated as separate ridge-classification samples (Hoerl & Kennard, 1970); labels only fit classifiers. Five-fold evaluation on pretraining images groups each image's representations and averages fold accuracies. Official-train/test fitting/evaluation assesses unseen images. All conditions share samples, splits, feature processing, and classifiers. Accuracy is in percent; differences are in percentage points (pp). Independent runs are statistical units; shared-model protocols/stages are repeated measurements. Within-run differences yield across-run means and paired 95% t intervals; the primary endpoint also uses a two-sided sign test.

The natural-image DINOv2-style configuration (Caron et al., 2021; Oquab et al., 2023; Zhou et al., 2021) uses DINO's image-level base loss, Image BERT Pre-Training with Online Tokenizer (iBOT)'s patch-level auxiliary loss A, and KoLeo R on encoder class-token (CLS) representations (enabled weight 0.1). Heads are shared as in iBOT; original DINOv2 separates image/patch heads. P removes only A's encoder gradient; N disables A while preserving matched masked inputs, providing a DINO base-loss reference.

Tiny ImageNet uses a randomly initialized ViT-S/16 with a shared head: eight paired runs per condition, 48 trajectories total. Splits, resolution, and schedules match; separate encoder/head clipping preserves F/P head-update matching. Training lasts 100 epochs, saving every 20 epochs. Ridge classifiers use frozen target-encoder CLS features; a training-data validation subset selects coefficients and compares stages. After verifying all 48 final checkpoints, fixed validation-selected coefficients determine accuracy on an independent held-out set under a common protocol.

4 Experimental Evidence Across Training Conditions

Experiments use Ubuntu 24.04 LTS, Python 3.11.15, PyTorch 2.11.0 (CUDA 12.8), and NumPy 2.1.3, with NVIDIA GeForce RTX 4090 and NVIDIA RTX PRO 6000 Blackwell Server Edition GPUs.

4.1 Regularization Effects under Matched Gradient Intervention

Paired comparisons of the six CUB conditions test whether regularization effects depend on direct auxiliary encoder updates (Table 1).

Table 1: Linear evaluation results of the Caltech–UCSD Birds-200-2011 main experiment.

	Five-fold linear evaluation on pretraining images			Linear evaluation on images unseen during pretraining		
Condition	0	1	δ	0	1	δ
N	12.80	5.95	-6.85	14.43	6.91	-7.52
F	4.02	6.00	+1.98	5.07	7.48	+2.40
P	7.60	5.90	-1.70	8.71	6.80	-1.91
D	$+3.67\ [2.99, 4.36]$			$+4.31\ [2.94, 5.68]$		
G	-6.80			-6.96		

Note: Means over six independent training runs at 20 epochs; both evaluations use the same models. N disables the auxiliary loss; F retains its direct encoder gradient; P removes only that gradient. 0/1: Gram off/on. Accuracy: %; effects: percentage points. δ_X, D, and G: regularization effect in X, F/P regularization interaction, and F1-versus-N0 combination effect. Brackets: paired 95% t intervals.

At the prespecified CUB20 five-fold endpoint, Gram improves F but worsens P in all six runs (Table 1; two-sided sign-test $p = 0.03125$, Table 9). Same-state predictor-gradient matching links this reversal to direct auxiliary encoder updates.

From mean initialization accuracy of 4.10%, N0 reaches 12.80% while F0 remains at 4.02% (Appendix F.3); gains over F0 therefore require comparison against the base-loss reference.

Reversal persists on unseen images in all six runs, but F1 trails N0 (Table 1). These evaluations are repeated measurements, not independent runs. Ordinary N/F switching also removes predictor updates, yielding $\delta_F - \delta_N = 9.92$ pp, interval $[9.04, 10.80]$. N/F identifies opposite signs of regularization and combination effects; F/P isolates update-path dependence. Their ratio is not a mechanistic contribution fraction.

4.2 Training Budget, Weight, and Regularizer Comparisons

Table 2: Extended comparisons of the four F/P conditions.

Setting	α	δ_F	δ_P	D	$D > 0$
CUB Gram, 75 ep.	1	$+2.26\ [-0.81, 5.32]$	$-2.57\ [-3.25, -1.89]$	$+4.83\ [1.36, 8.31]$	5/6
Pet Gram, 121 ep.	1	$+2.14\ [0.16, 4.12]$	$-0.03\ [-1.20, 1.15]$	$+2.16\ [-0.42, 4.75]$	5/6
CUB cov, 20 ep.	1	$+1.19\ [0.22, 2.16]$	$-2.72\ [-3.34, -2.11]$	$+3.91\ [2.46, 5.37]$	6/6
CUB Gram, 20 ep.	0.25	$+1.82\ [1.00, 2.63]$	$-1.07\ [-1.44, -0.70]$	$+2.89\ [2.04, 3.73]$	6/6

Note: Caltech–UCSD Birds-200-2011 (CUB)/Oxford-IIIT Pet (Pet): bird/pet datasets. F/P retains/removes direct auxiliary encoder gradients; α: full auxiliary-loss weight; cov: covariance regularizer. δ_F, δ_P: regularization effects under F, P; D: interaction. Effects: means and paired 95% t intervals (percentage points); final column: runs with $D > 0$/total independent runs. cov shares unregularized baselines with 20-epoch Gram at $\alpha = 1$; all four $\alpha = 0.25$ conditions are retrained. All rows evaluate images unseen during pretraining. Epochs count completed full epochs; exact steps: Appendix A.

At CUB75, five of six runs have $D > 0$, but only four have $\delta_F > 0$, whose interval crosses zero (Table 13). Interaction changes by $+0.52$ pp from 20 epochs (95% interval $[-1.82, 2.86]$), remaining positive on average with less consistent reversal (sign-test $p = 0.21875$).

Table 3: Tiny ImageNet held-out evaluation at 100 epochs.

Condition	Accuracy (%)
N0	30.63
N1	30.36
F0	32.05
F1	31.55
P0	30.69
P1	30.64

Comparison	Mean	95% interval
δ_N	-0.27	$[-0.62, 0.08]$
δ_F	-0.50	$[-0.95, -0.05]$
δ_P	-0.05	$[-0.35, 0.25]$
D	-0.454	$[-0.910, 0.002]$
G	$+0.92$	$[0.49, 1.34]$
$Y_{F,0} - Y_{P,0}$	$+1.36$	$[1.07, 1.66]$
$Y_{F,1} - Y_{P,1}$	$+0.91$	$[0.67, 1.14]$

Note: Left: mean accuracy (%) over eight independent training runs. Right: mean paired differences and 95% t intervals (percentage points). N disables the auxiliary loss; F retains its direct encoder gradient; P removes only that gradient. 0/1: regularization off/on. $Y_{X,b}$: accuracy for condition X, regularization state b; δ_X: corresponding regularization effect; D: F/P regularization interaction; G: F1-versus-N0 combination effect. The D interval retains three decimal places.

Across three increasing encoder-gradient coefficients, mean regularization effects rise from -2.57 through $+0.86$ to $+2.26$ pp, strictly increasing in four of six runs; the midpoint interval crosses zero ($[-1.57, 3.28]$; Figure 5, Table 15).

At $\alpha = 0.25$, all six runs retain reversal (Tables 2, 12), although interaction decreases by 1.43 pp relative to $\alpha = 1$ (paired interval $[-2.37, -0.49]$). The combination remains below the base-loss reference: $G = -7.46$ pp, interval $[-7.83, -7.10]$.

At 27900 steps, Pet's four F/P-by-regularization conditions yield positive mean interaction, but its interval crosses zero (Tables 2, 14).

Cov also reverses in all six runs (Tables 2, 11); Figure 2 compares regularizers and weights.

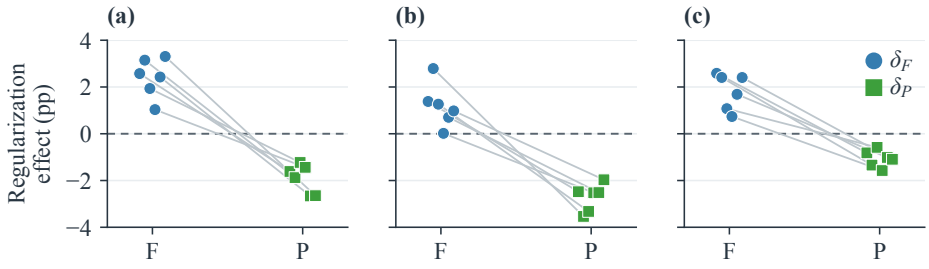


Figure 2: Regularization effects: official Caltech–UCSD Birds-200-2011 test set, 20 epochs. (a) Gram, $\alpha = 1$; (b) covariance regularizer, $\alpha = 1$; (c) Gram, $\alpha = 0.25$. Blue circles (δ_F)/green squares (δ_P): retain/remove direct auxiliary encoder gradients. Thin gray lines pair within-run effects; horizontal offsets separate overlaps. Panels: six independent runs, identical vertical scales; gray dashed lines: zero.

Despite reversal, cov's combination effect is negative in all six runs: $G = -8.17$ pp, interval $[-9.12, -7.23]$.

4.3 Regularization Effects in the Natural-Image Configuration

DINOv2-style Tiny ImageNet tests whether component gains correspond to full-combination performance (Table 3).

At 100 epochs, held-out KoLeo effects under F are negative in seven of eight runs, while combination effects are positive in all eight (Tables 3, 20). Thus, KoLeo lowers F accuracy on average despite the full combination exceeding the base-loss reference. Against CUB's $\delta_F > 0, G < 0$, this configuration gives $\delta_F < 0, G > 0$: the effects can disagree in either direction (Figure 3).

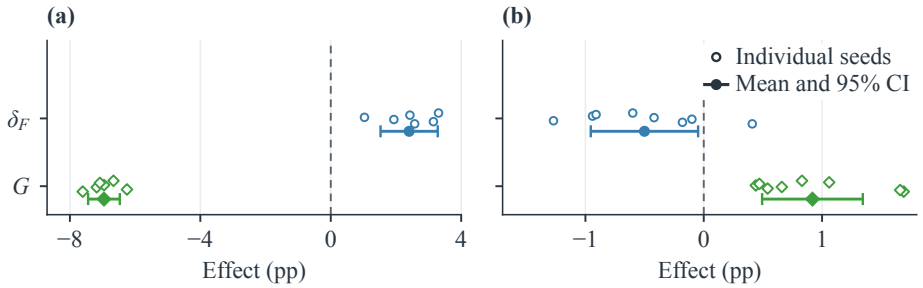


Figure 3: Regularization/combination effects. (a) Caltech–UCSD Birds-200-2011 (20 epochs)/Gram; (b) Tiny ImageNet/Kozachenko–Leonenko regularizer. Blue circles/green diamonds: regularization δ_F/combination G. Open/filled markers: independent runs/means; horizontal bars: paired 95% t intervals. δ_F baseline: F0 (all auxiliary gradients, regularization off); G baseline: N0 (base loss only). Panel-specific horizontal scales span respective effects; gray dashed lines: zero.

F exceeds P in all eight runs both with and without KoLeo (Table 3). Direct auxiliary encoder-update benefits and KoLeo's negative mean effect under F therefore require separate tests.

Interaction is negative in six runs, positive in one, and zero in one. The exact two-sided sign-flip test gives raw $p = 0.046875$ (Table 20; Appendix E), but the paired 95% interval crosses zero ($D = -0.454$ pp, $[-0.910, 0.002]$). This differs from CUB's consistent reversals and falls below the prespecified 1 pp practical-effect threshold in magnitude.

Tiny ImageNet stage comparisons share a validation set (Figure 4(b)); held-out evaluation assesses the 100-epoch endpoint.

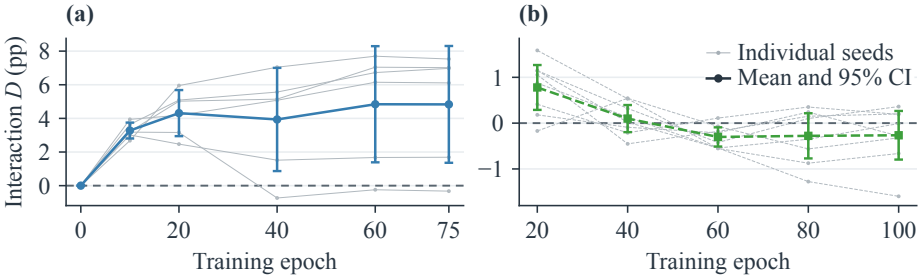


Figure 4: Interactions across training stages: (a) Caltech–UCSD Birds-200-2011; (b) Tiny ImageNet. D: regularization-effect difference retaining/removing direct auxiliary encoder gradients. Thin gray lines: within-run paired checkpoints. Mean interactions: (a) blue circles/solid line; (b) green squares/dashed line. Bars: paired 95% t intervals; horizontal gray dashed lines: zero. Panel-specific vertical scales; comparisons within panels.

Mean interactions at 20/40/60/80/100 epochs are $+0.78/+0.10/-0.30/-0.28/-0.26$ pp. The paired 100-versus-20-epoch change is -1.04 pp (95% interval $[-1.84, -0.25]$). Means shift from positive to slightly negative, with greater between-run variation later; 80- and 100-epoch intervals are $[-0.77, 0.22]$ and $[-0.80, 0.27]$. Extended CUB training instead retains positive mean interaction with lower directional consistency. Early interactions inform stage diagnosis; final effects inform component selection at the target budget.

5 Interpretation, Diagnostic Uses, and Scope

5.1 Gradient Paths and Representation Geometry

Shared parameters let state-centered residual losses affect state means. With predictor and target encoder fixed, ordinary gradient flow gives:

$$\left.\frac{dq}{dt}\right|_A = -\alpha J_q J_r^\top \nabla_r A, \qquad J_q = \frac{\partial q}{\partial \theta}, \quad J_r = \frac{\partial r}{\partial \theta}. \tag{5}$$

Here, $q = MH_\theta$ denotes encoder state means, $r = CP_{\theta,\phi}$ prediction residuals, and H_θ and $P_{\theta,\phi}$ encoder and predictor output tensors. $M = \mathbf{1}\mathbf{1}^\top/4$ and $C = I - M$ project onto state means and centered states; J_q and J_r are encoder-parameter Jacobians, t gradient-flow time, and $|_A$ the auxiliary-loss contribution. Although $A(P + \Delta P; T) = A(P; T)$ whenever $C\Delta P = 0$, shared parameters generally yield nonzero $J_q J_r^\top$.

Jacobian coupling (Jacot et al., 2018) identifies a pathway, not its accuracy effect: state means omit some class information, and Adam with decoupled weight decay (AdamW) (Loshchilov & Hutter, 2019) adds momentum and preconditioning. This local result establishes only that output-space separation need not separate parameter updates (Appendix G).

Table 4: Representation geometry and linear evaluation in the Caltech–UCSD Birds-200-2011 main experiment.

Condition	Base prediction loss	Total variance	Effective rank	Linear accuracy (%)
N0	0.1139	47.752	5.05	12.80
N1	0.0083	1.338	98.80	5.95
F0	0.0163	0.877	3.69	4.02
F1	0.0116	1.413	90.01	6.00
P0	0.2405	181.968	2.17	7.60
P1	0.0115	1.267	96.96	5.90

Note: Means of target-encoder representation diagnostics across six independent runs at 20 epochs; accuracy uses five-fold linear evaluation on pretraining images. N disables auxiliary loss; F retains its direct encoder gradient; P removes only that gradient. 0/1 denotes regularization off/on.

Gram raises effective rank under F, P, and N, yet accuracy improves only under F (Table 4). Alongside prior work (Ziyin et al., 2022; Bordes et al., 2022; Huang et al., 2024a), these results require task evaluation of geometric improvements under the corresponding auxiliary encoder updates.

5.2 Diagnostic Uses

After changing auxiliary tasks, sharing, or stop-gradient, four F/P-by-regularization conditions test update benefits and D; N0/N1 assess combination and base-configuration gains. Independent runs at the target budget should resolve imprecision and validate practical effects, including KoLeo removal. Transfer requires θ/ϕ partitions, auxiliary gradients to both, and same-state non-encoder-update checks covering normalization and gradient post-processing.

5.3 Evaluation Checks and Scope

Appendix C documents same-state gradient/update, initialization/resumption, and endpoint checks. Classifier refitting, prediction recomputation, and selected feature re-extraction agree; label shuffling approaches chance (Appendix H). Appendix I extends base-reference training. Attribution remains specific to the reported objectives, backgrounds, gradient controls, and budgets; variation across runs and configurations limits extrapolation.

6 Conclusion

Matched gradient intervention and a base-loss reference separate regularization gains, combination gains, and dependence on direct auxiliary encoder updates. CUB20 shows consistent Gram/covariance reversals; Gram retains reversal at both auxiliary weights, with different interaction magnitudes. In DINOv2-style held-out evaluation at 100 epochs, auxiliary encoder updates help with or without KoLeo, yet KoLeo lowers mean F accuracy while the full combination beats the base reference. These opposite-sign patterns require distinct references for component and combination selection, accounting for objectives, gradient recipients, and training budgets.

AI USE STATEMENT

In this work, we used generative AI tools to assist with mathematical proofs by suggesting proof ideas and completing intermediate steps, and to translate manuscript text between Chinese and English. We did not use generative AI tools for experimental design or execution, or for generating synthetic datasets. Additionally, we used generative AI tools to improve the clarity and readability of the writing and to assist with searching for and identifying relevant literature. We reviewed the AI-assisted proofs and verified the mathematical arguments step by step. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study uses existing image datasets for self-supervised training and offline evaluation. Our conclusions are limited to the data, models, training budgets, and evaluation protocols reported in this paper. Linear classification accuracy and representation geometry metrics do not constitute guarantees of safety or fairness in real-world deployment. The sources of the datasets and open-source implementations used in this study are documented in the main text, appendix, and supplementary materials. Use of these resources is subject to their original licenses and terms of use.

REPRODUCIBILITY STATEMENT

Section 3 describes the objective function, gradient-matched interventions, and effect definitions. Appendices A–D document the experimental matrix, data construction, loss functions, training settings, and verification of matching in both gradients and actual updates. Appendix E describes linear evaluation, data splits, and statistical calculations. Appendix F reports the results of individual independent training runs, and Appendix G provides the derivations. Appendices H and I provide evaluation checks and baseline training trajectories, respectively. The accompanying anonymous supplementary materials include configurations and result records, saved predictions, scripts for recomputing results, and usage instructions. These materials explicitly distinguish archived training checks from result recomputation that can be performed on a GPU.

REFERENCES

Gabriel Y Arteaga, Marius Aasan, Rwiddhi Chakraborty, Martine Hjelkrem-Tan, Thalles Santos Silva, Michael Kampffmeyer, and Adín Ramírez Rivera. Why prototypes collapse: Diagnosing and preventing partial collapse in prototypical self-supervised learning. In *International Conference on Learning Representations*, volume 2026, pp. 134907–134930, 2026.

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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

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

Florian Bordes, Randall Balestriero, Quentin Garrido, Adrien Bardes, and Pascal Vincent. Guillotine regularization: Why removing layers is needed to improve generalization in self-supervised learning. *arXiv preprint arXiv:2206.13378*, 2022.

Vivien Cabannes, Bobak Kiani, Randall Balestriero, Yann LeCun, and Alberto Bietti. The SSL interplay: Augmentations, inductive bias, and generalization. In *International conference on machine learning*, pp. 3252–3298. PMLR, 2023.

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.

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, 2020.

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.

Jingyi Cui, Hongwei Wen, and Yisen Wang. An augmentation-aware theory for self-supervised contrastive learning. *arXiv preprint arXiv:2505.22196*, 2025.

Siran Dai, Qianqian Xu, Peisong Wen, Yang Liu, and Qingming Huang. Exploring structural degradation in dense representations for self-supervised learning. *Advances in Neural Information Processing Systems*, 38:16715–16764, 2026.

Alexandre Devillers and Mathieu Lefort. EquiMod: An equivariance module to improve visual instance discrimination. In *International Conference on Learning Representations*, 2023.

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*, 2021. URL https://arxiv.org/abs/2010.11929.

Aleksandr Ermolov, Aliaksandr Siarohin, Enver Sangineto, and Nicu Sebe. Whitening for self-supervised representation learning. In *International conference on machine learning*, pp. 3015–3024. PMLR, 2021.

Quentin Garrido, Laurent Najman, and Yann LeCun. Self-supervised learning of split invariant equivariant representations. *arXiv preprint arXiv:2302.10283*, 2023.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

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.

Manu Srinath Halvagal, Axel Laborieux, and Friedemann Zenke. Implicit variance regularization in non-contrastive SSL. *Advances in neural information processing systems*, 36:63409–63436, 2023.

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. URL https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634.

Hanxun Huang, Ricardo Campello, Sarah Erfani, Xingjun Ma, Michael E Houle, and James Bailey. LDReg: local dimensionality regularized self-supervised learning. In *International Conference on Learning Representations*, volume 2024, pp. 16993–17018, 2024a.

Lei Huang, Yunhao Ni, Xi Weng, Rao Muhammad Anwer, Salman Khan, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Understanding whitening loss in self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 46(12):9479–9492, 2024b.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Wooseong Jeong and Kuk-Jin Yoon. Selective task group updates for multi-task optimization. In *International Conference on Learning Representations*, volume 2025, pp. 95046–95079, 2025.

Junguang Jiang, Baixu Chen, Junwei Pan, Ximei Wang, Dapeng Liu, Jie Jiang, and Mingsheng Long. ForkMerge: Mitigating negative transfer in auxiliary-task learning. *Advances in neural information processing systems*, 36:30367–30389, 2023.

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

Joohyung Lee, Changhun Kim, Hyunsu Kim, Kwanhyung Lee, and Juho Lee. Soft equivariance regularization for invariant self-supervised learning. *arXiv preprint arXiv:2603.06693*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Shentong Mo and Shengbang Tong. Connecting joint-embedding predictive architecture with contrastive self-supervised learning. *Advances in neural information processing systems*, 37:2348–2377, 2024.

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.

Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3498–3505. IEEE, 2012.

Jean Ponce, Basile Terver, Martial Hebert, and Michael Arbel. Dual perspectives on non-contrastive self-supervised learning. In *International Conference on Learning Representations*, volume 2026, pp. 31160–31181, 2026.

James B Simon, Maksis Knutins, Liu Ziyin, Daniel Geisz, Abraham J Fetterman, and Joshua Albrecht. On the stepwise nature of self-supervised learning. In *International Conference on Machine Learning*, pp. 31852–31876. PMLR, 2023.

Chenxin Tao, Honghui Wang, Xizhou Zhu, Jiahua Dong, Shiji Song, Gao Huang, and Jifeng Dai. Exploring the equivalence of siamese self-supervised learning via a unified gradient framework. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14411–14420. IEEE, 2022.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *International Conference on Machine Learning*, pp. 10268–10278. PMLR, 2021.

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.

Ladyna Wittscher. A survey on design choices for self-supervised learning in computer vision. *Artificial Intelligence Review*, 59(4):112, 2026.

Mingqi Wu, Qiang Sun, and Archer Yang. PCA++: How uniformity induces robustness to background noise in contrastive learning. *Advances in Neural Information Processing Systems*, 38: 131085–131132, 2026.

Ziyang Wu, Jingyuan Zhang, Druv Pai, XuDong Wang, Chandan Singh, Jianwei Yang, Jianfeng Gao, and Yi Ma. Simplifying DINO via coding rate regularization. *arXiv preprint arXiv:2502.10385*, 2025.

Yihao Xue, Eric Gan, Jiayi Ni, Siddharth Joshi, and Baharan Mirzasoleiman. Investigating the benefits of projection head for representation learning. In *International Conference on Learning Representations*, volume 2024, pp. 38101–38123, 2024.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33: 5824–5836, 2020.

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.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021.

Zhijian Zhuo, Yifei Wang, Jinwen Ma, and Yisen Wang. Towards a unified theoretical understanding of non-contrastive learning via rank differential mechanism. *arXiv preprint arXiv:2303.02387*, 2023.

Liu Ziyin, Ekdeep Singh Lubana, Masahito Ueda, and Hidenori Tanaka. What shapes the loss landscape of self-supervised learning? *arXiv preprint arXiv:2210.00638*, 2022.

A Experimental Matrix and Pairing Relationships

To clarify the statistical unit of each comparison, this section summarizes the experimental conditions, training endpoints, and pairing and reuse of models. The remainder of this appendix provides details of the training implementation, evaluation protocols, results of individual independent training runs, gradient-path derivations, and evaluation sensitivity checks. All effects follow the definitions in Eq. (4); accuracy is expressed as a percentage and differences in percentage points (pp). Table 5 lists the training conditions, numbers of independent runs, and endpoint steps for each comparison.

Table 5: Experimental matrix.

Comparison	Conditions	Runs	Endpoint steps
CUB/Gram	N0, N1, F0, F1, P0, P1	6	7440
CUB/longer training	F0, F1, P0, P1	6	27900
CUB/encoder gradient coefficient	$\beta \in \{0, 0.5, 1\}$, regularizer on/off	6	27900
CUB/weight of the full auxiliary loss	$\alpha = 0.25$, F0, F1, P0, P1	6	7440
CUB/cov	Fcov, Pcov, reusing F0, P0	6	7440
Pet/Gram	F0, F1, P0, P1	6	27900
Tiny ImageNet/KoLeo	N0, N1, F0, F1, P0, P1	8	140600
CUB/base-learning check	Extended N0 training	2	55800

Note: Caltech–UCSD Birds-200-2011 (CUB) and Oxford-IIIT Pet (Pet) are bird and pet datasets, respectively. Runs denote the numbers of independent training runs in each comparison, and endpoint steps denote optimizer updates; different checkpoints from the same training trajectory belong to the same training run. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. α is the weight of the full auxiliary loss, β is the encoder gradient coefficient for the auxiliary gradient, cov is the covariance regularizer, and KoLeo is the Kozachenko–Leonenko regularizer.

The five-fold linear evaluation and official test-set evaluation on CUB use the same models to measure class readout on pretraining images and images unseen during pretraining, respectively. Comparisons across training stages and gradient coefficients also preserve the corresponding relationships: the longer-training comparison uses later checkpoints from the original training trajectories; in the encoder gradient coefficient comparison, the endpoints $\beta = 0$ and $\beta = 1$ reuse the P and F models, respectively. The base-learning check continues the N0 models from independent training runs 1 and 2 to analyze learning under the base objective.

The regularizer and auxiliary-weight comparisons retain their respective baselines and pairing relationships. The cov comparison adds 12 training trajectories with the regularizer enabled, reuses F0/P0 from the Gram experiment, and shares N0 with Gram to compute the combination effect. The comparison of the weight of the full auxiliary loss retrains the four F/P conditions at $\alpha = 0.25$, yielding 24 trajectories in total, while the N0 reference remains the base model from the corresponding training run.

Statistical analyses use independent training runs as the unit of analysis and preserve the pairing and model-reuse relationships above. Different evaluation protocols for the same models, different checkpoints from the same trajectory, and comparisons sharing baselines are all handled according to their corresponding relationships.

B Data Construction, Losses, and Training Settings for Predictive SSL

To specify the auxiliary objective and geometric constraints in predictive SSL, this section describes background-state generation, loss computation, and shared training settings in turn.

B.1 Images and Background States

The background construction specifies input changes across states while preserving the image foreground, and consistent sampling ensures data pairing across training conditions.

SSL training on CUB uses 5952 images, and linear evaluation on the official training and test sets uses 5994 and 5794 images, respectively. SSL training on Pet uses 3664 images; the linear classifier is fitted on 3673 valid trainval images and evaluated on 3662 test images. SSL training uses only images from the training split, and class labels are used only for linear evaluation after training.

Specifically, the foreground of the original image is preserved using the segmentation mask, and the background is replaced. Within each prediction direction, the four states of the same image share the foreground, crop, and input transformations, differing only in background. The coefficient vectors of the four states are

$$v_1 = (1, 1, 1), \quad v_2 = (1, -1, -1), \quad v_3 = (-1, 1, -1), \quad v_4 = (-1, -1, 1).$$

Here, v_s is the three-dimensional coefficient vector of background state s, with $s \in \{1, 2, 3, 4\}$. The coefficients for the source and target directions are $Q(k\pi/8)v_s$ and $Q(\pi/4)Q(k\pi/8)v_s$, respectively, where $k \in \{0, \ldots, 15\}$ is the phase index, and $Q(\omega)$ rotates by ω in the plane spanned by the first two coordinates while leaving the third coordinate unchanged. The two prediction directions use separate predictors.

The background is synthesized from three sinusoidal spatial basis functions with frequencies $(5, 0)$, $(0, 5)$, and $(3, 4)$, respectively. Each basis function is mean-centered on a 224×224 grid and normalized to unit root mean square. The background is centered at

$$0.5 + \frac{0.18}{\sqrt{3}} \sum_{j=1}^{3} c_j B_j,$$

where $c \in \mathbb{R}^3$ is the coefficient vector for the corresponding state and direction, and B_j is the jth normalized basis function. Smooth noise with a root mean square of 0.012 is then added, pixel values are clipped to $[0, 1]$, and the same grayscale background is replicated across the three color channels. The smooth noise is generated by bicubic interpolation of a 28×28 Gaussian grid, then mean-centered, normalized, and scaled to the specified root mean square. The noise random stream is jointly determined by the training run, image, training epoch, direction, absolute phase, and state index to ensure that paired conditions receive identical data.

Each step contains 16 images corresponding to the 16 phases, with four background states per image. The four image groups for Gram are randomly generated and fixed within each independent training run; grouping is independent of class, and each group contains four images per batch. Conditions corresponding to N, F, P and the regularizer on/off settings share the image order, grouping, phases, and masks in the same paired training run.

B.2 State-Residual Alignment and Geometric Regularizers

The auxiliary loss aligns residuals across background states, whereas Gram and cov constrain the group-mean geometry and covariance structure of online representations, respectively.

For each prediction direction, centering along the state dimension first gives

$$\widetilde{P}_s = P_s - \tfrac{1}{4} \sum_{u=1}^{4} P_u, \qquad \widetilde{T}_s = T_s - \tfrac{1}{4} \sum_{u=1}^{4} T_u.$$

Here, P_s and T_s are the prediction and target tensors for state s, respectively; $\widetilde{P}_s$ and $\widetilde{T}_s$ are the state-centered residuals, and u indexes the sum over states. The full tensors are denoted by P_{mistd} and T_{mistd}, where the indices m, i, s, t, d denote target mask, image, background state, token, and feature dimension, respectively; indices not involved in the current operation are omitted from the equations. A whitening matrix along the state dimension is then estimated from the target residuals:

$$W(T_*) = \sum_{j=1}^{3} \frac{u_j u_j^\top}{\sqrt{\max(\nu_j, \tau)}}, \qquad \tau = \max(10^{-4} \nu_{\max}, 10^{-12}). \tag{6}$$

Here, $T_* \in \mathbb{R}^{4\times L_*}$ is the target-residual matrix arranged by state, L_* is the number of columns after merging the remaining dimensions, and $K = T_* T_*^\top / L_*$; ν_j and u_j are the three largest eigenvalues of K and their corresponding unit eigenvectors, respectively; $\nu_{\max}$ is the largest eigenvalue, τ is the eigenvalue floor, and W is the whitening matrix acting along the state dimension. The auxiliary loss consists of instance-wise and aggregated components. The instance-wise component is $\mathrm{SmoothL1}(W\widetilde{P}, W\widetilde{T})$, where matrix multiplication acts along the state dimension and the loss is averaged over all elements. For the aggregated component, the state-centered residuals are averaged successively over the mask, image, and token dimensions to obtain a 4×384 matrix; a whitening matrix is then estimated independently from the aggregated target residuals using Eq. (6), and the corresponding SmoothL1 is computed. The two components are added to obtain the auxiliary loss for one direction, and the two directions are averaged with equal weights to form A. Stop-gradient is applied to the target representations and both whitening matrices, and eigendecomposition and the residual loss use 32-bit floating-point precision (32-bit floating point, FP32).

The geometric regularizers act directly on online encoder representations before the predictor. To compute Gram, representations are first averaged over the states, masks, and tokens of each image, and four group means are then computed using the fixed random grouping. The group means are centered and substituted into Eq. (2), with a numerical floor of 10^{-12} for the trace denominator. The regularizers in the two directions are averaged with equal weights, and the enabled weight is 1.

The ungrouped covariance regularizer is

$$R_{\mathrm{cov}} = \frac{1}{d}\sum_j \left[\max\left(0, 1 - \frac{\sigma_j}{s_0}\right)\right]^2 + \frac{1}{d(d-1)}\sum_{j\neq k}\left(\frac{V_{jk}}{\sigma_j \sigma_k}\right)^2. \tag{7}$$

Here, $Z \in \mathbb{R}^{n\times d}$ contains centered token features, n is sample count, and $d = 384$ feature dimension; $V = Z^\top Z/(n-1)$ is sample covariance, $\sigma_j = \sqrt{V_{jj} + \epsilon}$ the stabilized standard deviation of dimension j, ϵ a numerical stabilizer, s_0 the median initial per-dimension feature standard deviation, and j, k dimension indices. The terms constrain the variance floor and off-diagonal correlations.

For cov, the mask, image, state, and token dimensions of the online encoder token representations are merged into a sample dimension, and the centered covariance is computed in FP32. In Eq. (7), $s_0 = 0.79908318346261$, $\epsilon = \max(10^{-12}, 10^{-8} s_0^2)$, and the enabled weight is 3.002213185071846. Calibration uses three independent engineering initializations, with 16 batches per initialization; the regularization weight is determined by the ratio of the median encoder gradient norms of Gram and cov, and s_0 is the median per-dimension standard deviation of features at initialization. After calibration, all formal training runs use the same values, and the cov losses in the two directions are averaged with equal weights.

B.3 Model and Optimization Schedule

All paired conditions use the same model architecture and optimization schedule to separate the effects of the auxiliary gradient path and the regularizer on/off setting.

The training implementation is based on commit `52c1ae9` of the official I-JEPA code. The online encoder is a randomly initialized ViT-S/16; the target encoder has the same architecture and is initialized from the online encoder. Each of the two prediction directions uses a predictor with depth 12 and dimension 384. Table 6 summarizes the shared training settings, with epochs calculated using 16 original images per step.

The 20-epoch CUB results are taken from the 7440-step checkpoint of the same 27900-step schedule, and longer training continues with the original schedule. The comparison of the weight of the full auxiliary loss retrains the four conditions at $\alpha = 0.25$ from the initial state of the corresponding training run, keeping the data, Gram weight, and schedule unchanged; this coefficient scales both the instance-wise and aggregated components of the auxiliary loss and applies to both the encoder and predictors. By contrast, the β comparison keeps $\alpha = 1$ and predictor gradients unchanged, varying only the auxiliary gradient received by the encoder.

Table 6: Training and optimization settings for predictive self-supervised learning.

Item	Setting
Encoder	ViT-S/16; 12 layers, 6 attention heads, feature dimension 384
Input	224×224; crop area $[0.30, 1]$, aspect ratio $[0.75, 4/3]$
Input transformations	Normalization with ImageNet mean and standard deviation; horizontal flipping, color jitter, and Gaussian blur disabled
Context mask	1 mask, area ratio $[0.85, 1]$
Target masks	4 masks, area ratio $[0.15, 0.20]$, aspect ratio $[0.75, 1.50]$
Mask matching	The four states of each image share masks; context and target regions are disjoint, with at least 10 tokens retained
Optimizer and precision	AdamW with the official parameter grouping; BF16 forward pass, FP32 for sensitive losses; no gradient clipping
Learning rate	Warmup from 6.25×10^{-6} to 3.125×10^{-5}, followed by cosine decay to 10^{-6}
CUB warmup and schedule	3720 warmup steps; full schedule of 27900 steps
Weight decay and EMA	Weight decay from 0.04 to 0.4; target-encoder EMA coefficient from 0.996 to 1
CUB training endpoints	20 epochs correspond to 7440 steps and 75 epochs to 27900 steps, with 372 steps per epoch
Pet training endpoint	229 steps per epoch and 27900 steps in total, comprising 121 complete epochs and, in epoch 122, 191 steps

Note: Oxford-IIIT Pet (Pet) and Caltech–UCSD Birds-200-2011 (CUB) are pet and bird datasets, respectively. EMA denotes exponential moving average, BF16 denotes the bfloat16 (16-bit floating-point) format, and FP32 denotes the 32-bit floating-point format. AdamW denotes Adam with decoupled weight decay.

C F/P Gradient Intervention and Implementation Verification

To restrict the scope of the F/P intervention to the direct gradient from the auxiliary loss to the encoder, this section describes the implementation and verification of shared backpropagation, actual updates, and training resumption.

C.1 Backpropagation at the Same Computational State

F/P preserves the gradients of non-encoder parameters through shared backpropagation and replaces only encoder gradients.

Let $B = L_{\text{base}} + \lambda_b R$. Each matching check fixes the online and target networks, optimizer, precision scaler, inputs, masks, and random-number states, and performs the following steps in order:

1. Perform a shared forward pass to compute B and A, retaining the computation graph.
2. Compute and cache $\nabla_\theta B$; the cache also preserves the `None` state for parameters that produce no gradient.
3. Clear accumulated gradients and perform a shared full backward pass on $B + \alpha A$ to obtain gradients for the encoder and all other parameters.
4. F retains the full gradients. P replaces encoder gradients with the cached $\nabla_\theta B$, retaining the gradients of all other parameters from the full backward pass. For a general β condition, encoder gradients are set to a linear combination of the cached and full gradients with coefficients $1 - \beta$ and β, respectively.
5. Apply precision handling, gradient clipping, and the optimizer update as in the original implementation, then update the target-network EMA.

When precision scaling is used, the cached and full backward passes use the same scaling state, and gradients are replaced at a common scale; preserving the `None` state ensures consistent optimizer

handling of parameters not involved in differentiation. The CUB branch then performs the original AdamW update, while the natural-image branch clips encoder and projection-head gradients separately, so that the projection head's clipping factor depends only on its own gradients.

C.2 Matching Checks and Training Resumption

Matching checks verify the scope of the intervention at the same computational state, while resumption checks verify continuity of the training state and schedule.

Verification compares, in order, shared forward values, encoder gradient sources, raw gradients with respect to ϕ, and parameters and optimizer states after the update. Actual single-step updates execute F and P separately from the same checkpoint; diverged training trajectories are not used for checks at the same computational state. During formal training, each condition evolves independently, and resumption continues the learning-rate, weight-decay, and EMA clocks from the original step count.

CUB completed 144 sets of gradient comparisons on each of the 4090 and Pro6000. Actual-update checks started from a frozen state at step 1860 of a historical training run and covered two precisions, two regularizer states, two independent engineering runs, and two hardware platforms, yielding 16 F/P comparisons. The maximum absolute differences in both updated predictor parameters and their optimizer states were 0, FP32 encoder gradients passed the prespecified numerical tolerances, and none of the listed checks encountered overflow or skipped optimizer steps.

The extended experiments follow the corresponding matching requirements. Each of the six formal cov training runs completed an 8-step check covering the initial state, equivalence at entry with no regularizer, F/P forward passes and predictor updates, encoder gradients, and replay after resumption. Each of the six independent training runs at $\alpha = 0.25$ completed a 14-step check and verified the coefficient scaling of the full auxiliary loss and its gradients. Final evaluation covered the 12 new cov trajectories and the 24 new trajectories in the lower-weight experiment, respectively; results recomputed from saved predictions agreed with the corresponding accuracy records.

The DINOv2-style experiment checks single-step projection-head updates after separate clipping of the encoder and projection-head groups. After all 48 checkpoints at 100 epochs passed identity and endpoint-step verification, held-out evaluation was performed uniformly.

C.3 Basis for Matching Shared Backpropagation and Actual Updates

Shared backpropagation controls numerical differences in finite-precision differentiation, and verification of actual updates further checks gradient postprocessing and optimizer states.

In exact arithmetic, computing the gradients of the base and auxiliary terms with respect to ϕ separately and then adding them is equivalent to a single backward pass on the total loss; under finite precision, intermediate rounding and aggregation order may produce different numerical values. F/P therefore obtains the gradients of ϕ through the same full backward pass and then replaces encoder gradients according to the condition.

Table 7 illustrates this requirement through numerical comparisons in an earlier implementation. The check uses four frozen model states from a set of historical training runs and one fixed real batch, sets the modules to evaluation mode, and compares repeated joint differentiation, ordinary joint backpropagation, and separate differentiation on the same forward graph. Each state restores its corresponding precision scaler, and the models, optimizers, and EMA remain frozen throughout the check. The initialization and endpoint states come from an earlier implementation, whose training results are treated separately from the matching experiments for the six paired training runs in the main text.

The relative difference is $\|g_{\text{split}} - g_{\text{joint}}\|_2 / \|g_{\text{joint}}\|_2$. Here, g_{split} is the predictor gradient obtained by adding separately computed gradients, and g_{joint} is the predictor gradient obtained by joint backpropagation.

The four states and two precisions yield eight comparisons. Predictor gradient differences are zero both between repeated joint differentiations and between joint differentiation and ordinary joint backpropagation, whereas differences between separate differentiation and joint backpropagation vary with state and precision. These values reflect implementation differences on one fixed batch;

Table 7: Comparison of predictor gradients in an earlier implementation.

Frozen state	BF16 relative difference (%)	BF16 cosine	FP32 relative difference (%)
F0, initialization	7.443684	0.998038	0.000377
F0, endpoint	56.951696	0.880747	0.004763
P0, endpoint	22.175157	0.975507	0.001069
F1, endpoint	4.105888	0.999161	0.000472

Note: Relative differences are expressed as percentages, and cosine denotes the cosine similarity between the two gradients. BF16 denotes the bfloat16 (16-bit floating-point) format, and FP32 denotes the 32-bit floating-point format. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively, and 0/1 indicates that the regularizer is disabled/enabled.

formal F/P training uses shared backpropagation and additionally verifies the actual optimizer updates.

Matching of actual updates executes F and P separately from the same model and optimizer states. In the 16 paired CUB comparisons described above, the maximum absolute difference in updated predictor parameters and optimizer states is 0. The natural-image branch applies norm clipping separately to the encoder and projection head: if all parameters share a clipping factor, changes in the encoder gradient norm also change the magnitude of projection-head updates; separate group clipping makes the projection head's clipping factor depend only on its own gradients. Matching verification therefore covers forward values, raw gradients, and post-update states.

In addition to single-step matching, the natural-image experiments compare 4 consecutive steps against resumption across processes after 2+2 steps using independent engineering runs before formal execution, covering N0, F1, and P1. The model, center, and random states are bitwise identical; optimizer states on the two 4090s are bitwise identical, while the maximum absolute difference in the N0 optimizer state on the Pro6000 is 2.78×10^{-17}, satisfying an absolute tolerance of 10^{-12} and a relative tolerance of 10^{-7}. The short-sequence checks verify the handling of training states and schedules during resumption, while the 48 trajectories of 100 epochs are verified through their checkpoint identities, step counts, and endpoint evaluation records.

D Training and Data Settings for the Natural-Image Experiments

To describe the implementation of the six-condition comparison in the natural-image setting, this section presents the data splits, view construction, model and training objectives, and optimization and pairing settings.

D.1 Data Splits and Inputs

The data splits and input construction distinguish training, validation, and held-out evaluation while keeping the input rules consistent across the six conditions. Tiny ImageNet contains 200 classes. Before training, exact duplicate images were checked using SHA-256 hashes of the red, green, and blue pixel channels, and the official validation set was reserved for the final held-out evaluation. From the original training split, images exactly duplicated in that set and exact duplicates with conflicting labels were removed, excluding 47 images in total. The remaining original training images in each class were then sorted by the SHA-256 hash of a fixed string prefix and the file path; 450 images were selected for training, and the rest for validation. The resulting training set contains 90000 images, the validation set 9953 images, and the held-out set the 10000 images in the official validation set; there are no exact pixel duplicates across the three retained sets.

Each training image produces two 224×224 global crops and eight 96×96 local crops, with area ratios of $[0.32, 1]$ and $[0.05, 0.32]$, respectively. The original images have a resolution of 64×64 and are upsampled to the specified sizes after cropping. Global views use a 14×14 patch grid, a

masking probability of 0.5, and a mask ratio sampled from $[0.1, 0.5]$. The six conditions share the same input and mask generation rules, and N also retains masked inputs.

D.2 Model, Objectives, and Optimization

The six conditions share the same model architecture, forward computation, and optimization schedule, with differences determined by auxiliary-gradient control and whether the regularizer is enabled. The implementation is based on commit `7764ea0` of the official DINOv2 code and uses standard scaled dot-product attention. The encoder is a randomly initialized ViT-S/16 with dimension 384 and no register tokens. The image-level and patch-level objectives share a three-layer projection head with a hidden dimension of 2048, a bottleneck dimension of 256, and 8192 prototypes, without batch normalization. The target network is initialized from the online network, and both the encoder and projection head are updated through EMA.

The base objective is the DINO cross-view cross-entropy, and the auxiliary objective is the iBOT masked patch-level objective. For the image-level loss, the eight local views and two target global views form 16 pairs, with two additional pairs formed by crossing the global views; the loss is normalized over all 18 pairs. KoLeo is applied separately to the encoder CLS representations of the two global views, summed, and weighted by 0.1 when enabled. F, P, and N use the same forward computation and center update rules; the gradients in N come only from the base loss and the corresponding regularizer, whereas P retains the iBOT gradients to the shared projection head.

Training runs for 100 epochs with 1406 steps per epoch. Detailed optimization settings and intermediate checkpoints are listed in Table 8.

Table 8: Optimization settings for the DINOv2-style experiments.

Item	Setting
Batch size, optimizer, and precision	Batch size 64; AdamW, momentum coefficients $(0.9, 0.999)$; BF16
Learning rate	Peak 2×10^{-4}, 10-epoch warmup, cosine decay to 10^{-6}
Weight decay and EMA	Cosine schedules: weight decay from 0.04 to 0.4, EMA coefficient from 0.996 to 1
Temperature and centering	Online temperature 0.1; target temperature warmed up from 0.04 to 0.07 over 30 epochs; center momentum 0.9
Network settings	drop path 0.1; initial LayerScale value 10^{-5}
Parameter-group learning rates	Layer-wise decay 0.9; patch-embedding learning-rate multiplier 0.2; final projection-head layer frozen for 1 epoch
Gradient clipping	Encoder and projection head clipped separately to a norm of 3
Intermediate checkpoints	Epochs 20, 40, 60, 80, 100; corresponding to steps 28120, 56240, 84360, 112480, 140600, respectively

Note: EMA denotes exponential moving average, and BF16 denotes the bfloat16 (16-bit floating-point) format. The optimizer is Adam with decoupled weight decay (AdamW).

Conditions in the same paired training run share initialization, sample order, and the random streams for augmentation and masking. Checkpoints save the online network, target network, optimizer, and center states together; training resumed from a checkpoint restores the original 100-epoch schedule according to the cumulative number of optimization steps.

E Linear Evaluation and Statistical Computation

To compare training conditions under a consistent evaluation protocol, this section describes the extraction of frozen representations, fitting of linear classifiers, the evaluation roles of the datasets, and the computation of paired effects and uncertainty.

E.1 Frozen-Representation Evaluation on CUB and Pet

For both CUB and Pet, representations are evaluated by linear classification with the target encoder frozen, and data splits are organized by original image. For each image, 384-dimensional representations are extracted for the two directions and four background states, averaging over target masks and tokens within each state to obtain eight representations in total. Each representation corresponding to a direction and state is treated as one classification sample, and accuracy is computed over these samples; all representations of the same original image are kept together when constructing the splits.

The CUB five-fold evaluation uses 5952 pretraining images, assigning folds according to image identifiers and a fixed randomized evaluation-split rule; all states and directions of the same image are assigned to the same fold. Within each fold, per-dimension means and standard deviations are estimated only on the training side, standard deviations below 10^{-12} are set to 1, and the resulting standardization transform is then applied to the evaluation side. The ridge classifier is solved as follows:

$$\widehat{W} = (\widetilde{X}^\top \widetilde{X} + \rho J)^{-1} \widetilde{X}^\top Y, \qquad \rho = 10^{-4} \frac{\mathrm{tr}(X^\top X)}{384}, \quad J = \mathrm{diag}(1, \ldots, 1, 0). \tag{8}$$

Here, X is the standardized training feature matrix, Y is the one-hot class-label matrix, $\widetilde{X} = [X, \mathbf{1}]$ is the matrix augmented with an intercept column, $\widehat{W}$ contains the classifier coefficients, ρ is the ridge penalty coefficient, J is a diagonal matrix whose final diagonal entry is 0 and whose remaining entries are 1, diag denotes the construction of a diagonal matrix, and $\mathbf{1}$ is a column vector of 1s. The intercept is unpenalized, the solution is computed in 64-bit floating-point precision, and the predicted class is the class with the highest score. The final five-fold accuracy is the arithmetic mean of the fold accuracies.

Evaluation on the official test set uses the standardization parameters estimated on the training side and fits the classifier according to Eq. (8) using all features from the official training images before applying it to the test images. The comparisons involving cov and $\alpha = 0.25$ follow this protocol. For Pet, the corresponding trainval/test images are used for fitting and evaluation, respectively, with 37-class one-hot targets; all other linear-classification settings remain the same.

E.2 Validation and Held-Out Evaluation on Tiny ImageNet

For Tiny ImageNet, the validation set is used to select the classifier coefficient and compare training stages, while an independent held-out set is used to evaluate the 100-epoch endpoint. Images are first resized to 256×256 using bicubic interpolation, center-cropped to 224×224, and normalized using ImageNet normalization, after which the 384-dimensional layer-normalized CLS representations of the target encoder are extracted. When fitting the classifier, feature means and standard deviations are estimated using only the 90000 training images, with standard deviations floored at 10^{-8}. The ridge classifier is solved as follows:

$$\widehat{W}_\eta = \left(X^\top X/n + \eta I\right)^{-1} X^\top (Y - \mathbf{1}\bar{y}^\top)/n.$$

Here, X is the standardized training feature matrix, Y is the one-hot class-label matrix, n is the number of training images, $\bar{y}$ is the mean of the training label vectors, η is the ridge coefficient, I is the identity matrix, $\mathbf{1}$ is a column vector of 1s, and $\widehat{W}_\eta$ contains the corresponding classifier coefficients. The classification scores are $x^\top \widehat{W}_\eta + \bar{y}^\top$, where x is the standardized feature column vector of an image. The solution is computed in 64-bit floating-point precision using symmetric eigendecomposition, with numerically negative eigenvalues clipped to 0. The ridge coefficient is selected from $10^{-6}, 10^{-5}, \ldots, 10^{3}$ according to validation accuracy, with ties resolved in favor of the larger value.

At each stage, the classifier is fitted using the training features from that stage, and its coefficient is selected on the validation set. At 100 epochs, the selected coefficient, standardization parameters, and classifier for each model are fixed and used to predict directly on the 10000 held-out images. Thus, changes across stages are computed only within the same validation set, and the held-out results are used separately for the 100-epoch endpoint evaluation.

E.3 Paired Intervals, Signs, and Changes across Stages

The statistical analysis uses independent training runs as its units to quantify the mean, uncertainty, and directional consistency of paired effects. δ_X, D, and G are first computed within each training run and then summarized across runs. The mean paired difference and its 95% interval are

$$\bar{d} = \frac{1}{n}\sum_i d_i, \qquad \bar{d} \pm t_{n-1,0.975}\frac{s_d}{\sqrt{n}}, \qquad s_d^2 = \frac{1}{n-1}\sum_i (d_i - \bar{d})^2.$$

Here, d_i is the paired difference in the ith independent training run, n is the number of independent training runs, $\bar{d}$ is the mean paired difference, s_d is the sample standard deviation, and $t_{n-1,0.975}$ is the 0.975 quantile, with $n-1$ degrees of freedom, of the Student t distribution. The intervals reflect uncertainty across independent training runs; the numbers of images, states, and folds are not treated as numbers of independent training runs. Comparisons between two training stages or auxiliary weights likewise first compute the change within the same paired training run and then construct intervals across runs.

Directional consistency is assessed using the sign test, with zero differences reported separately; the exact sign-flip test, which retains the magnitudes of the differences, is computed as follows:

$$p = 2^{-n}\sum_{\boldsymbol{\epsilon}} \mathbf{1}\left\{\left|n^{-1}\sum_i \epsilon_i d_i\right| \geq |\bar{d}|\right\}.$$

Here, $\epsilon_i \in \{-1, 1\}$ is the sign applied to the ith paired difference, $\boldsymbol{\epsilon}$ enumerates all 2^n sign combinations, $\mathbf{1}\{\cdot\}$ is the indicator function, and p is the probability value of the two-sided exact sign-flip test. The exact sign-flip test assumes sign exchangeability under the null hypothesis and differs from the sign test, which counts only positive and negative differences. The numerical implementation uses a tolerance of 10^{-12} pp for zero values and boundary comparisons, and all reported exact tests provide raw two-sided p values. Results across stages are analyzed using paired effects and intervals; the interaction effects on Tiny ImageNet are additionally assessed against a prespecified threshold of 1 pp for a practically meaningful effect.

F Results for Individual Training Runs and Comparisons across Stages

This section reports the results of all independent training runs by experimental configuration and evaluation protocol, and uses paired comparisons to describe changes in the effects across training stages. Table means are computed from the unrounded raw results, and regularization effects and interaction effects are recomputed according to Eq. (4). The different CUB evaluation protocols are reported separately, with the official test set used for the cov and lower-weight comparisons; Tiny ImageNet endpoint results use the held-out set, while stage results use the validation set.

F.1 The Main CUB Experiment and Two Extensions

We first report the two linear evaluations of the main CUB experiment and then compare paired effects with the covariance regularizer and a lower auxiliary weight; the five-fold results of the main experiment are shown in Table 9.

Table 9: Five-fold linear evaluation of the main experiment on Caltech–UCSD Birds-200-2011.

Run	N0	F0	F1	P0	P1	δ_F	δ_P	D	G
1	13.504	3.797	6.648	7.980	6.387	2.851	-1.593	4.444	-6.856
2	13.137	4.037	6.590	7.393	6.234	2.553	-1.159	3.712	-6.546
3	12.289	4.436	5.899	8.086	6.283	1.462	-1.802	3.265	-6.390
4	12.145	4.248	5.320	6.608	5.009	1.072	-1.599	2.671	-6.825
5	12.486	3.811	5.480	7.765	5.754	1.669	-2.011	3.680	-7.006
6	13.247	3.788	6.048	7.754	5.741	2.260	-2.013	4.273	-7.199
Mean	12.801	4.020	5.998	7.598	5.901	1.978	-1.696	3.674	-6.804

Note: Training lasts 20 epochs, with $\alpha = 1$; α is the weight of the full auxiliary loss. Accuracy is reported in %, and effects in percentage points. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. δ_X is the regularization effect under condition X, D is the F/P regularization interaction effect, and G is the combination effect of F1 relative to N0.

The official test-set results for the same models are shown in Table 10 to compare effects on images unseen during pretraining.

Table 10: Official test-set evaluation of the main experiment on Caltech–UCSD Birds-200-2011.

Run	N0	F0	F1	P0	P1	δ_F	δ_P	D	G
1	15.274	5.094	7.667	9.003	7.389	2.574	-1.614	4.188	-7.607
2	14.224	4.826	7.974	8.766	6.891	3.148	-1.875	5.022	-6.250
3	14.226	5.113	7.048	8.561	7.333	1.935	-1.228	3.163	-7.178
4	13.702	5.715	6.748	7.749	6.313	1.033	-1.437	2.470	-6.953
5	14.450	4.943	7.370	9.111	6.451	2.427	-2.660	5.087	-7.081
6	14.726	4.757	8.062	9.072	6.425	3.305	-2.647	5.952	-6.664
Mean	14.434	5.075	7.478	8.710	6.800	2.404	-1.910	4.314	-6.955

Note: Training lasts 20 epochs, with $\alpha = 1$; α is the weight of the full auxiliary loss. Accuracy is reported in %, and effects in percentage points. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. δ_X is the regularization effect under condition X, D is the F/P regularization interaction effect, and G is the combination effect of F1 relative to N0.

Table 11 reports the results after replacing Gram with cov while keeping the unregularized baselines and evaluation protocol unchanged.

Table 11: Official test-set results for the covariance regularizer on Caltech–UCSD Birds-200-2011.

Run	N0	F0	F1	P0	P1	δ_F	δ_P	D	G
1	15.274	5.094	6.474	9.003	6.524	1.381	-2.479	3.860	-8.800
2	14.224	4.826	7.616	8.766	5.230	2.790	-3.536	6.326	-6.608
3	14.226	5.113	6.377	8.561	5.238	1.264	-3.322	4.587	-7.849
4	13.702	5.715	5.732	7.749	5.225	0.017	-2.524	2.541	-7.969
5	14.450	4.943	5.646	9.111	6.595	0.703	-2.516	3.219	-8.804
6	14.726	4.757	5.734	9.072	7.104	0.977	-1.968	2.945	-8.992
Mean	14.434	5.075	6.263	8.710	5.986	1.189	-2.724	3.913	-8.170

Note: Training lasts 20 epochs; cov denotes the covariance regularizer. Here, F1 and P1 indicate that cov is enabled; F0, P0, and N0 are shared with the Gram comparison. Accuracy is reported in %, and effects in percentage points. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. δ_X is the regularization effect under condition X, D is the F/P regularization interaction effect, and G is the combination effect of F1 relative to N0.

After reducing the weight of the full auxiliary loss to $\alpha = 0.25$, the results for the four retrained conditions and their base-loss reference are shown in Table 12.

Table 12: Official test-set results with a lower auxiliary weight on Caltech–UCSD Birds-200-2011.

Run	N0	F0	F1	P0	P1	δ_F	δ_P	D	G
1	15.274	5.038	7.622	8.349	7.531	2.585	-0.818	3.402	-7.652
2	14.224	4.895	7.303	8.606	7.266	2.408	-1.340	3.747	-6.921
3	14.226	5.922	6.994	7.525	6.943	1.072	-0.582	1.655	-7.232
4	13.702	5.566	6.304	8.010	6.438	0.738	-1.573	2.311	-7.398
5	14.450	5.029	6.716	7.890	6.876	1.687	-1.014	2.701	-7.734
6	14.726	4.481	6.886	8.002	6.910	2.406	-1.092	3.497	-7.840
Mean	14.434	5.155	6.971	8.064	6.994	1.816	-1.070	2.886	-7.463

Note: Training lasts 20 epochs; the weight of the full auxiliary loss is $\alpha = 0.25$. N0 reuses the base-loss reference from the corresponding paired training run. Accuracy is reported in %, and effects in percentage points. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. δ_X is the regularization effect under condition X, D is the F/P regularization interaction effect, and G is the combination effect of F1 relative to N0.

Compared with the original weight, the interaction effect decreases in all six independent training runs at the lower weight, with a mean paired change of -1.428 pp and a 95% interval of $[-2.370, -0.487]$. The mean change in F0 accuracy is $+0.081$ pp, with an interval of $[-0.320, 0.481]$. These two comparisons reflect changes in the interaction magnitude and in the accuracy of the unregularized configuration, respectively.

F.2 Training Budget, Encoder Gradient Coefficient, and Cross-Dataset Comparisons

This subsection extends CUB training, transfers the experiment to the Pet dataset, and varies the encoder gradient coefficient to compare regularization effects under these conditions; the endpoint results for the longer CUB training are shown in Table 13.

The four-condition endpoint results for Pet are shown in Table 14, with the same pairing scheme as the F/P comparison on CUB.

Table 13: Official test-set results on Caltech–UCSD Birds-200-2011 (75 epochs).

Run	F0	F1	P0	P1	δ_F	δ_P	D
1	4.981	9.152	9.087	7.147	4.170	-1.940	6.110
2	4.833	9.223	9.359	6.733	4.390	-2.626	7.016
3	10.642	8.407	8.850	6.940	-2.235	-1.909	-0.326
4	7.909	7.288	8.617	6.306	-0.621	-2.311	1.689
5	5.128	8.610	9.376	5.883	3.482	-3.493	6.975
6	4.789	9.160	9.223	6.062	4.371	-3.161	7.531
Mean	6.381	8.640	9.085	6.512	2.260	-2.573	4.833

Note: The four conditions are paired within the same independent training run. Accuracy is reported in %, and effects in percentage points. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively, and 0/1 indicates that the regularizer is disabled/enabled. δ_F and δ_P are the regularization effects under F and P, respectively. D is the F/P regularization interaction effect.

Table 14: Official test-set results on Oxford-IIIT Pet at the 27900-step endpoint.

Run	F0	F1	P0	P1	δ_F	δ_P	D
1	13.408	15.982	14.982	14.118	2.574	-0.864	3.437
2	12.872	16.074	13.582	14.412	3.202	0.829	2.372
3	14.671	16.429	15.050	14.490	1.758	-0.560	2.318
4	13.179	16.955	14.220	15.125	3.775	0.905	2.871
5	17.238	15.773	13.906	15.009	-1.464	1.103	-2.567
6	13.073	16.057	14.852	13.278	2.983	-1.574	4.557
Mean	14.074	16.212	14.432	14.405	2.138	-0.027	2.165

Note: The four conditions are paired within the same independent training run. Accuracy is reported in %, and effects in percentage points. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively, and 0/1 indicates that the regularizer is disabled/enabled. δ_F and δ_P are the regularization effects under F and P, respectively. D is the F/P regularization interaction effect.

In the longer CUB training configuration, only the coefficient of the auxiliary-loss gradient to the encoder is varied; the resulting regularization effects are shown in Table 15.

Table 15: Regularization effects under different encoder gradient coefficients on Caltech–UCSD Birds-200-2011 (75 epochs).

Run	$\beta = 0$ (P)	$\beta = 0.5$	$\beta = 1$ (F)
1	-1.940	2.336	4.170
2	-2.626	2.695	4.390
3	-1.909	-1.210	-2.235
4	-2.311	-2.852	-0.621
5	-3.493	2.151	3.482
6	-3.161	2.011	4.371
Mean	-2.573	0.855	2.260

Note: Effects are reported in percentage points. The weight of the full auxiliary loss is $\alpha = 1$; β is the auxiliary gradient coefficient received by the encoder, while the auxiliary gradient coefficient received by the predictor remains 1. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively.

Under the respective fixed test protocols, the mean within-run paired change in the interaction effect on CUB from 20 to 75 epochs is $+0.519$ pp, with a 95% interval of $[-1.823, 2.860]$; the correspond-

ing mean change on Pet from 7440 to 27900 steps is -0.452 pp, with an interval of $[-2.396, 1.491]$. Figure 5 further shows the paired relationships among the three encoder gradient coefficients at the endpoint of the longer CUB training.

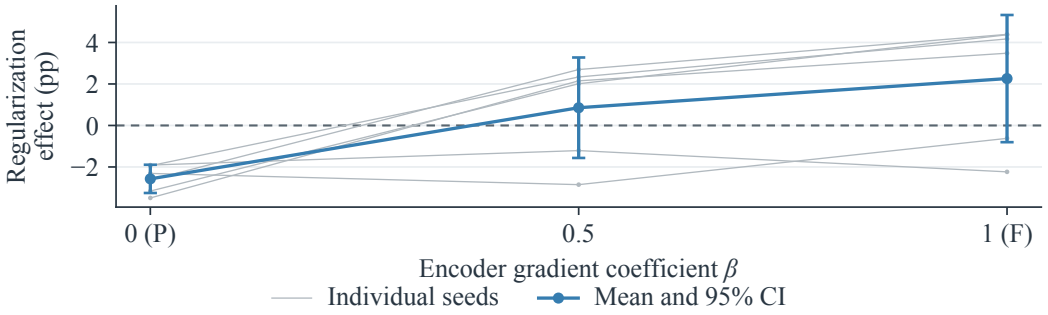


Figure 5: Encoder gradient coefficient and regularization effect on Caltech–UCSD Birds-200-2011 (75 epochs). Thin light-blue lines connect the three measured coefficients within the same paired training run; dark-blue dots and error bars show the mean regularization effect and paired 95% t confidence intervals, and the gray dashed line denotes zero effect. The coefficient $\beta = 0$ (P) removes the auxiliary encoder gradient, while $\beta = 1$ (F) retains the full auxiliary gradient. The connecting lines indicate pairing; values between the three coefficients were not measured.

F.3 Six-Condition Learning Trajectories and Effects across Stages

This subsection uses paired comparisons across stages for the same models to distinguish changes in mean effects from differences across training runs. Table 16 reports the accuracies of the six conditions in the main CUB experiment at matching stages: the conditions start from the same initialization within the same paired training run, each time point is evaluated using the established five-fold protocol, and the results are then averaged across six independent training runs. N0 provides a base-loss reference for learning, and the crossed comparison of F/P and the regularizer switch is used to analyze how direct encoder updates from the auxiliary loss affect the regularization effect.

Table 16: Five-fold linear-classification accuracy trajectories for the main experiment on Caltech–UCSD Birds-200-2011.

Condition	0	1	5	10	15	20
N0	4.099	4.251	8.870	11.311	12.036	12.801
N1	4.099	4.196	5.112	6.575	6.137	5.949
F0	4.099	4.088	4.088	4.112	4.138	4.020
F1	4.099	4.211	5.065	6.481	6.141	5.998
P0	4.099	4.308	6.975	6.978	7.182	7.598
P1	4.099	4.198	5.136	6.558	6.273	5.901

Note: Accuracy is reported in %. Columns indicate training epochs, with 372 steps per epoch; each column reports means over the same six independent training runs. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled.

Under this protocol, the mean interaction effects at the six stages are 0, 0.2330, 2.8171, 2.7889, 2.9111, and 3.6741 pp, respectively. Class readout under N0 improves with training; after training, F1 exhibits a positive regularization effect relative to F0 and a negative combination effect relative to N0. The stage trajectories are repeated measurements of the same models, and both accuracy and interaction effects at each stage use independent training runs as the statistical units.

To correspond to the evaluation on images unseen during pretraining in the main text, Table 17 lists stage results for the three encoder gradient coefficients on the official CUB test set, with the evaluation protocol held fixed across stages.

Table 17: Effects across stages on the official Caltech–UCSD Birds-200-2011 test set.

Epoch	δ_P	$\delta_{\beta=0.5}$	δ_F	D	$D > 0$	p
0	0.0000	0.0000	0.0000	0.0000	0/6	1
10	-0.4764	2.7899	2.8007	3.2771	6/6	0.03125
20	-1.9100	2.0409	2.4037	4.3137	6/6	0.03125
40	-2.4447	1.1905	1.4908	3.9355	5/6	0.21875
60	-2.6773	0.9320	2.1635	4.8409	5/6	0.21875
75	-2.5731	0.8551	2.2595	4.8326	5/6	0.21875

Note: Values are means over six independent training runs, with effects reported in percentage points. The weight of the full auxiliary loss is $\alpha = 1$, and β scales only the gradient from the auxiliary loss to the encoder. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively. δ is the regularization effect under the corresponding condition, and D is the F/P interaction effect. The $D > 0$ column gives the number of independent training runs with a positive interaction divided by the total number of runs; the final column reports the raw p value of the two-sided exact sign test.

The corresponding stage results for Pet are shown in Table 18, likewise using the fixed evaluation protocol for that dataset at every stage.

Table 18: Effects across stages on the official Oxford-IIIT Pet test set.

Completed epochs	δ_F	δ_P	D	$D > 0$	p
0	0.0000	0.0000	0.0000	0/6	1
32	2.2637	-0.3533	2.6170	6/6	0.03125
64	1.5383	-0.3897	1.9280	5/6	0.21875
121	2.1379	-0.0267	2.1647	5/6	0.21875

Note: Values are means over six independent training runs, with effects reported in percentage points. The epoch column records the number of fully completed epochs, with 229 steps per epoch; the checkpoints listed at 32, 64, and 121 epochs occur at steps 112, 224, and 191 of the following epoch, respectively, corresponding to 7440, 14880, and 27900 updates. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively. δ_F and δ_P are the regularization effects under F and P, respectively, and D is the interaction effect. The $D > 0$ column gives the number of independent training runs with a positive interaction divided by the total number of runs; the final column reports the raw p value of the two-sided exact sign test.

The mean interaction effect on CUB changes from 4.3137 pp at 20 epochs to 4.8326 pp at 75 epochs, while the sample standard deviation across training runs increases from 1.3057 to 3.3094 pp. On Pet, the mean interaction effect changes from 2.6170 pp at 7440 steps to 2.1647 pp at the endpoint, with an endpoint sample standard deviation of 2.4615 pp. At the endpoints of both experiments, 5/6 training runs have positive interaction effects, while the individual endpoint results also include runs with effects in the opposite direction. Changes across stages therefore need to be analyzed jointly in terms of mean effects, dispersion across runs, and sign consistency, corresponding to the distinction in the main text between interaction magnitude and differences across training runs.

F.4 Held-Out Results at 100 Epochs and Stage Results on the Validation Set

This subsection separately reports the 100-epoch held-out results for Tiny ImageNet and stage results on the same validation set, distinguishing endpoint evaluation from changes in effects during training; the held-out accuracies for the six conditions are shown in Table 19.

Paired effects are computed from the six-condition accuracies within the same training run, with the results shown in Table 20.

Table 19: Held-out accuracies on Tiny ImageNet.

Run	N0	N1	F0	F1	P0	P1
1	30.42	30.61	31.70	32.11	30.85	30.70
2	30.29	30.45	32.13	31.95	30.38	31.10
3	30.57	29.89	32.38	31.11	30.85	30.58
4	31.00	30.78	31.76	31.66	30.69	30.59
5	30.90	30.43	31.76	31.34	30.59	30.77
6	30.95	29.96	32.36	31.42	30.55	30.47
7	30.34	30.42	32.31	31.40	30.73	30.52
8	30.56	30.33	31.99	31.39	30.85	30.38
Mean	30.63	30.36	32.05	31.55	30.69	30.64

Note: Results correspond to 100 epochs, with accuracy reported in %. Each row contains all six conditions from the same paired training run. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled.

Table 20: Paired effects in the held-out evaluation on Tiny ImageNet.

Run	δ_F	δ_P	D	G	F0−P0	F1−P1
1	0.41	-0.15	0.56	1.69	0.85	1.41
2	-0.18	0.72	-0.90	1.66	1.75	0.85
3	-1.27	-0.27	-1.00	0.54	1.53	0.53
4	-0.10	-0.10	0.00	0.66	1.07	1.07
5	-0.42	0.18	-0.60	0.44	1.17	0.57
6	-0.94	-0.08	-0.86	0.47	1.81	0.95
7	-0.91	-0.21	-0.70	1.06	1.58	0.88
8	-0.60	-0.47	-0.13	0.83	1.14	1.01
Mean	-0.50	-0.05	-0.45	0.92	1.36	0.91

Note: Results correspond to 100 epochs, with effects reported in percentage points. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this gradient; 0/1 indicates that the regularizer is disabled/enabled. δ_X is the regularization effect under condition X, D is the F/P regularization interaction effect, and G is the combination effect of F1 relative to N0. The final two columns give the F/P accuracy differences without and with KoLeo, respectively. Kozachenko–Leonenko (KoLeo) is the regularizer used in this table.

In the 4th independent training run, D is zero; the residual of order 10^{-15} pp produced by floating-point arithmetic is treated as zero. Of the remaining seven runs, six are negative and one is positive. The two-sided exact sign-flip test for D gives $p = 0.046875$, and the 95% interval for the mean is $[-0.910, 0.002]$ pp.

Changes across stages are evaluated on the same validation set. Table 21 reports the mean effects, intervals, and distribution of interaction directions at each stage.

Based on paired results from the same eight independent training runs, the mean change in the validation-set interaction effect from 20 to 100 epochs is -1.044 pp, with a paired 95% interval of $[-1.836, -0.251]$. This difference reflects only the change across stages on the same validation set; the 100-epoch held-out results remain separately used for endpoint evaluation.

G Derivations for Gradient Interventions and Representation Changes

To clarify the relationship among where the auxiliary loss is applied, its direct gradients, and its effects on subsequent training, this section derives the local parameter coupling relations used in the

Table 21: Comparisons across stages on the Tiny ImageNet validation set.

Epoch	δ_F	δ_P	D (95% interval)	Positive/negative/zero
20	+0.014	-0.765	$+0.779\ [0.289, 1.268]$	7/1/0
40	-0.575	-0.674	$+0.099\ [-0.197, 0.395]$	5/3/0
60	-0.539	-0.237	$-0.301\ [-0.511, -0.092]$	1/7/0
80	-0.458	-0.181	$-0.278\ [-0.770, 0.215]$	4/4/0
100	-0.421	-0.156	$-0.265\ [-0.796, 0.266]$	3/5/0

Note: The same validation set is used at each stage; effects are means over eight independent training runs, reported in percentage points. Square brackets give paired 95% intervals, and the final column gives the numbers of independent training runs with positive/negative/zero interaction effects. F/P retains/removes the direct gradient from the auxiliary loss to the encoder, respectively. δ_F and δ_P are the regularization effects under F and P, respectively. D is the F/P regularization interaction effect.

main text and explains the different roles of representation scale in linear evaluation and training. Ordinary gradient flow is used to analyze update paths, whereas accuracy and its interaction effects are measured by the paired training experiments in the main text.

G.1 State Centering and Coupling Induced by Shared Parameters

To explain why state centering in output space can still be accompanied by changes in state means, this subsection first presents the projection relations and then analyzes the coupling induced by shared parameters. Define projection matrices along the four background states as

$$M = \tfrac{1}{4}\mathbf{1}\mathbf{1}^\top, \qquad C = I - M, \qquad M^2 = M, \quad C^2 = C, \quad MC = CM = 0.$$

Here, M is the mean projection over the four background states, C is the state-centering projection, I is the identity matrix of order four, and $\mathbf{1}$ is the four-dimensional all-ones vector. With the target encoder and target-side whitening matrix fixed, both the per-instance and aggregated auxiliary losses in Appendix B depend through CP on the prediction tensor P. Therefore, for prediction perturbations that are constant across states,

$$A(P + \Delta P; T) = A(P; T), \qquad M\nabla_P A = 0. \tag{9}$$

Here, P and T are the prediction and target tensors, respectively, and ΔP is a perturbation that is constant across states and satisfies $C\Delta P = 0$; $\nabla_P A$ is the gradient of the auxiliary loss with respect to the prediction tensor. Taking the directional derivative of the first equality along any $\Delta P = M\Delta P$ gives the second equality. Target-side whitening uses stop-gradient, and its numerical floor does not alter this property with respect to the prediction tensor.

At a local computational state with ϕ and the target encoder fixed, the contribution of the auxiliary loss αA to ordinary gradient flow satisfies

$$\dot{\theta}\Big|_A = -\alpha J_r^\top \nabla_r A, \tag{10}$$

$$\dot{q}|_A = J_q\, \dot{\theta}\Big|_A = -\alpha J_q J_r^\top \nabla_r A. \tag{11}$$

Here, $\dot{\theta}|_A$ and $\dot{q}|_A$ are the contributions of the auxiliary loss to parameter velocity and the rate of change of the representation mean, respectively; $q = MH_\theta$ and $r = CP_{\theta,\phi}$ are the vectorized state means and prediction residuals, respectively, and $J_q = \partial q/\partial\theta$ and $J_r = \partial r/\partial\theta$ are the corresponding Jacobians. The orthogonality of the output projections, $MC = 0$, and the parameter-Jacobian inner product, $J_q J_r^\top$, concern different objects. The latter depends on the encoder, predictor, and current parameters, and shared parameters can affect both q and r. Thus, the output translation invariance in Eq. (9) does not preclude changes in state means. $J_q J_r^\top = 0$ is a sufficient condition for the current auxiliary update to preserve q; for a specific gradient, $J_q J_r^\top \nabla_r A = 0$ suffices for the contribution at that instant to be zero.

The derivation above uses only the finite-dimensional chain rule and does not require the Jacobians to remain constant throughout training. The state mean is a computable component of the representation, whereas class evaluation uses the full state features. This derivation therefore motivates direct examination of the encoder update path; the direction of the accuracy change is still determined by training and evaluation.

G.2 Paths of Subsequent Influence Retained by Condition P

To explain how the auxiliary predictor updates retained by P can affect subsequent encoder training, the following compares P and N under the same regularization status and initialization. P retains the auxiliary-loss updates to ϕ; under ordinary gradient flow, both conditions satisfy $\dot{\theta} = -\nabla_\theta B$, but their velocity difference for ϕ is

$$\dot{\phi}_P(0) - \dot{\phi}_N(0) = -\alpha \nabla_\phi A.$$

Here, the subscripts P and N denote the two training conditions, $t = 0$ is the shared initial state, a dot denotes the first derivative with respect to gradient-flow time, and $B = L_{\text{base}} + \lambda_b R$. The two conditions have identical initial encoder and target-network velocities. In a sufficiently smooth local region, differentiating the encoder velocity once more gives

$$\ddot{\theta}_P(0) - \ddot{\theta}_N(0) = \alpha \big[\partial_\phi \nabla_\theta B\big] \nabla_\phi A.$$

Here, a double dot denotes the second derivative with respect to gradient-flow time, and $\partial_\phi \nabla_\theta B$ is the derivative of the encoder gradient with respect to the head parameters. The representation difference between the two conditions therefore satisfies

$$q_P(t) - q_N(t) = \frac{\alpha t^2}{2} J_q \big[\partial_\phi \nabla_\theta B\big] \nabla_\phi A + O(t^3). \tag{12}$$

Here, $q(\theta)$ is a differentiable representation quantity that depends only on the encoder, J_q is its Jacobian with respect to encoder parameters, t is gradient-flow time, and $O(t^3)$ denotes the remainder of order three and higher. All quantities on the right-hand side are evaluated at the shared initial state. Because R acts only on the encoder, $\partial_\phi \nabla_\theta B = \partial_\phi \nabla_\theta L_{\text{base}}$. Equation (12) thus describes a path in which the auxiliary loss first changes ϕ and then affects the encoder through the base objective; at locations such as the piecewise boundaries of SmoothL1, this expression must be interpreted in light of the corresponding local differentiability.

For example, take scalar parameters with $L_{\text{base}} = (\phi\theta - 1)^2/2$ and $A = (\phi\theta)^2/2$, start from $\theta = \phi = 1$, and set $q = \theta$ and $R = 0$. The expression above gives $q_P(t) - q_N(t) = \alpha t^2/2 + O(t^3)$. Even with identical initial encoder velocities, the subsequent trajectories of P and N can diverge. Accordingly, the F/P comparison in the main text measures the full training changes caused by removing the direct auxiliary gradient to the encoder, including the subsequent evolution of parameters, the optimizer, and the target network.

G.3 Standardized Linear Evaluation and Representation Scale

To distinguish the relationship between geometric scale and class readout, this subsection first describes the properties of standardized linear evaluation under per-dimension scale transformations and then analyzes regularization gradients during training. Total representation variance and effective rank characterize the geometric state, whereas linear evaluation directly measures class readout performance. In the following, the training and test split is fixed, the features undergo an invertible affine transformation in each dimension, and the training standard deviations before and after the transformation are assumed to exceed the numerical floor. Using the respective training-side means and standard deviations before and after the transformation, the resulting standardized features satisfy

$$x'_j = \operatorname{sign}(a_j) x_j.$$

Here, x_j and x'_j are the separately standardized features in dimension j before and after the transformation; the transformation of the raw features is $z'_j = a_j z_j + b_j$, where $a_j \neq 0$ is the scale, b_j is the translation, and sign is the sign function. Let S denote the corresponding diagonal sign matrix,

so that $S^\top S = I$. If $X' = XS$ and $W' = S^\top W$, then $X'W' = XW$ and $\|W'\|_F = \|W\|_F$; moreover, $\mathrm{tr}(X'^\top X') = \mathrm{tr}(X^\top X)$, so the ridge coefficient in Appendix E also remains the same. Therefore, with exact solutions and refitting of the standardization parameters and classifier, the two sets of features yield identical predictions. This property applies to invertible affine transformations in each dimension; general linear transformations change the geometry of the ℓ_2 penalty.

This property of standardized evaluation does not eliminate the influence of scale on training optimization. In the region where the trace denominator in Eq. (2) exceeds the numerical floor, the Gram regularizer is invariant to positive uniform scaling, and hence

$$\langle U, \nabla_U R(U) \rangle = 0, \qquad \nabla_U R(\kappa U) = \kappa^{-1} \nabla_U R(U).$$

Here, U is the centered group-mean matrix, $\kappa > 0$ is a scale factor satisfying $R(\kappa U) = R(U)$, $\langle \cdot, \cdot \rangle$ is the Frobenius inner product, and $\nabla_U R$ is the gradient of the regularizer with respect to this matrix. After mapping through the parameter update, the output change is $\Delta U \simeq -\gamma \lambda_b J_U J_U^\top \nabla_U R$, whose radial component depends on $J_U J_U^\top$. Here, γ is the gradient step size, and J_U is the Jacobian of the group means with respect to encoder parameters. Thus, radial orthogonality in output space cannot directly determine the scale change after a parameter update; the roles of scale in standardized evaluation and training optimization must be analyzed separately, and both geometric metrics and task evaluation must be retained.

H Checks and Sensitivity Analyses for Linear Evaluation

To describe the scope of the evaluation checks and the sensitivity of the results to readout settings, this section presents checks of the data and training execution, comparisons of pooling methods and ridge coefficients, and the correspondence between geometric metrics and task evaluation. The supplementary CUB checks use the official training split for fitting and the official test split for evaluation; the fixed-coefficient checks on Tiny ImageNet use the 20-epoch validation results. Each check retains its corresponding models, independent training runs, and stages, serving respectively to recompute results, analyze sensitivity to evaluation settings, and compare representation geometry.

H.1 Checks of Data, Classifiers, and Training Execution

To verify consistency in the data splits, classifier computations, and training execution, this subsection summarizes the objects, scope, and results of the checks. The primary CUB evaluation uses target-token pooled features from the frozen target encoder: the 5994 images in the official training split are used to fit the standardization parameters and the 200-class ridge regression classifier, and the 5794 test images are used for scoring; the five-fold evaluation on the original 5952 pretraining images is recomputed independently. All background states and prediction directions of the same image remain in the same split. The detailed check results are reported in Table 22.

Accuracy after label shuffling is close to chance for 200 classes, and independent classifier refitting and prediction recomputation both yield consistent results. Feature re-extraction verifies actual forward-pass results at selected checkpoints, whereas the 216 metadata checks verify labels and splits; their coverage differs. The training execution checks verify the number of updates, optimizer and EMA calls, and log continuity; learning progress is analyzed together with the curves in Appendix I.

H.2 CUB Pooling Methods, Ridge Coefficients, and an External Positive Control

To analyze the effects of readout settings on accuracy, this subsection compares pooling methods and ridge-coefficient selection and uses an external positive control to check the class readout capability of the same data and evaluation procedure. The fixed ridge-coefficient ratio is 10^{-4}, consistent with Appendix E; the alternative setting validates the ratio grid $\{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$ on the training side, selects a ratio based on validation results, and then scores on the official test set. Table 23 reports results for both whole-image pooling and target-token pooling; all SSL models are from the first independent training run.

Table 22: Supplementary checks of the evaluation and training procedures.

Object checked	Coverage	Result
Labels and data splits	216 evaluation metadata entries from long-run CUB training	Consistent with official labels and splits
Five-fold ridge regression refitting	Independent training run 1, six conditions at 20 epochs	Maximum accuracy difference of approximately 1.78×10^{-15} percentage points
Independent ridge regression prediction	Six selected checkpoints	Identical predictions
Label shuffling	The same six checkpoints	Accuracy of 0.4423%–0.5566%
Feature re-extraction by forward passes	Selected checkpoints at initialization, F without/with Gram, P without Gram, and N	Corresponding features match the original cache
Training records	Newly executed training segments of 36 long-run CUB trajectories	All 825840 records are continuous; optimizer and EMA call checks pass; 0 gradient-scaler decreases

Note: Caltech–UCSD Birds-200-2011 (CUB) is the bird dataset used in this table. EMA denotes exponential moving average.

The external positive control uses an ImageNet-supervised pretrained 18-layer residual network (ResNet18)(He et al., 2016), extracting 512-dimensional whole-image pooled features to test whether the same rendered data and classification evaluation can yield high accuracy. The comparison must account for differences in architecture, supervision, and pretraining budget between this control and the ViT-S/16 trained from scratch in this work.

Table 23: Official test accuracy of selected models.

Model/budget	Features	Fixed coefficient	Validation-selected
F1, 75 epochs	Whole-image pooling	9.9176	10.0837
F1, 75 epochs	Target-token pooling	9.1517	9.3049
F0, 75 epochs	Whole-image pooling	5.9415	5.8164
F0, 75 epochs	Target-token pooling	4.9814	5.0375
P0, 75 epochs	Whole-image pooling	10.7223	10.7223
P0, 75 epochs	Target-token pooling	9.0870	9.1905
External supervised pretrained ResNet18	Whole-image pooling	54.3105	54.4270
N0, 20 epochs	Whole-image pooling	17.9108	17.7144
N0, 20 epochs	Target-token pooling	15.2744	14.9702

Note: Accuracy is reported in %. All self-supervised learning models are from the first independent training run; "validation-selected" means that the ridge regression coefficient is selected using training-side validation. Rows with different budgets separately check their respective evaluation settings. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this direct gradient. The 0/1 in condition identifiers denotes the regularizer being disabled/enabled. The 18-layer residual network (ResNet18) serves as an external positive control.

For the 75-epoch F conditions in the table, both pooling methods and both ridge-coefficient settings preserve the direction of increased accuracy when Gram is enabled. Under whole-image pooling, the gains with the fixed and validation-selected coefficients are 3.9761 and 4.2673 pp, respectively; under target-token pooling, they are 4.1703 and 4.2674 pp, respectively. The external positive control exceeds 54% under both coefficient settings, showing that this data construction and evaluation procedure permit substantially better class readout than the models trained from scratch in this work.

These sensitivity checks cover F0, F1, P0, and the base-loss reference from the first independent training run and directly establish the direction of the regularization effect under F for the listed readout settings. The full interaction D also involves P1, and its robustness across readout settings requires a four-condition comparison under the same protocol. The reversal of the regularization effect in the main text and its directional consistency across independent training runs are computed from the complete F/P matrix of six independent training runs under the prescribed readout protocol.

H.3 Fixed Ridge Coefficients and Stage-Specific Results on Tiny ImageNet

To test whether the direction of the early interaction depends on selecting ridge coefficients separately for each condition, this subsection recomputes the 20-epoch interaction effect using the same coefficient for all training conditions. Accuracies for ten candidate ridge coefficients were saved at this stage, and Table 24 reports the results for all candidates. The mean D is positive for every fixed coefficient, consistent with the direction obtained when coefficients are selected separately for each condition at 20 epochs; these comparisons reuse the same eight independent training runs and validation images.

Table 24: Sensitivity checks with fixed ridge coefficients on Tiny ImageNet.

Fixed η	Mean D	95% t interval across training runs	$D > 0$
10^{-6}	$+0.7573$	$[0.2800, 1.2347]$	7/8
10^{-5}	$+0.7460$	$[0.2672, 1.2248]$	7/8
10^{-4}	$+0.7636$	$[0.2845, 1.2427]$	7/8
10^{-3}	$+0.8264$	$[0.4163, 1.2365]$	8/8
10^{-2}	$+0.9156$	$[0.5323, 1.2988]$	8/8
10^{-1}	$+1.0575$	$[0.5780, 1.5370]$	8/8
10^{0}	$+1.0537$	$[0.7063, 1.4011]$	8/8
10^{1}	$+1.1981$	$[0.8389, 1.5573]$	8/8
10^{2}	$+1.4757$	$[1.0186, 1.9328]$	8/8
10^{3}	$+1.4556$	$[1.0347, 1.8765]$	8/8

Note: Effects are reported in percentage points; results are from the validation set at 20 epochs for eight independent training runs, and all rows reuse the same models. η is the fixed ridge coefficient, and D is the F/P regularization interaction effect. Intervals are paired 95% t confidence intervals; the last column gives the number of independent training runs with $D > 0$ divided by the total number of runs.

The fixed-coefficient check concerns the 20-epoch validation results and shows that the direction of the early interaction is maintained across the listed coefficients. The 100-epoch held-out evaluation in the main text uses the classifier selected on the validation set and frozen for each model; the results are reported in Table 20. The two evaluations correspond to different training stages and evaluation sets, serving respectively as an early-stage evaluation sensitivity analysis and an independent evaluation at the training endpoint.

H.4 Correspondence Between Geometric Metrics and Task Evaluation

To compare geometric changes with classification effects, this subsection presents representation diagnostics for the natural-image configuration and the main CUB experiment. Table 25 lists geometric metrics for the 20-epoch natural-image experiment, all computed on training-set CLS representations: effective rank is the exponential of the spectral entropy obtained by normalizing the nonnegative eigenvalues of the centered covariance, and total variance is the covariance trace. KoLeo

increases effective rank under N, F, and P, whereas the regularization effects at the corresponding stage are -0.525, $+0.014$, and -0.765 pp, respectively, indicating that geometric changes and task effects provide different information.

Table 25: Geometric metrics of the target encoder's training-set representations at 20 epochs.

Condition	Total variance without KoLeo	Total variance with KoLeo	Effective rank without KoLeo	Effective rank with KoLeo
N	154.61	363.00	34.83	190.99
F	171.09	369.04	31.62	186.35
P	166.81	366.23	36.95	185.81

Note: Values are means over eight independent training runs, using class-token representations. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this direct gradient. Kozachenko–Leonenko (KoLeo) is the regularizer used in this table.

Table 26 further reports the complete geometric diagnostics for all six conditions in the main CUB experiment. Total variance is the trace of the centered covariance of target representations, normalized prediction mean squared error is the pooled prediction mean squared error divided by the centered target energy, and prediction loss is the token-level SmoothL1 used in training. All metrics are computed on the same evaluation samples and splits and averaged with equal weight across six independent training runs.

Table 26: Endpoint representation diagnostics for matched gradient interventions.

Condition	Prediction loss	Total variance	Normalized prediction mean squared error	Effective rank
N0	.113892	47.751970	.445652	5.053518
N1	.008335	1.337963	.819307	98.802518
F0	.016255	.876632	3.302188	3.686810
F1	.011630	1.412973	.903599	90.010039
P0	.240487	181.968135	1.006451	2.165210
P1	.011479	1.266566	.929612	96.963515

Note: Values are means over six independent training runs on the same evaluation sample set and splits. Prediction loss is the token-level smooth L_1 loss (smooth L1 loss); normalized prediction mean squared error corresponds to pooled features. N disables the auxiliary loss, F retains its direct gradient to the encoder, and P removes only this direct gradient. The 0/1 in condition identifiers denotes the regularizer being disabled/enabled.

Gram increases effective rank under both F and P, but the regularization effects have opposite signs; F0 and P0 also exhibit different orderings in base prediction loss, total variance, and effective rank. Together with the corresponding linear classification results, the CUB and natural-image experiments show that changes in geometric objectives and the task effects of regularization must be matched by condition and evaluated separately.

I Learning Curves and Extended Training of the Base-Loss Reference

To supplement the analysis of learning progress under the base objective, this section describes the extended-training settings for N0 and analyzes changes across stages using accuracy and prediction loss from individual runs.

I.1 Training Settings and Evaluation Targets

To clarify the schedule and evaluation scope of extended training, this subsection describes how the models were resumed and the evaluation sets used. The base-learning check uses the N0 models from the first and second independent training runs: the first 75 epochs follow the original 27900-step schedule in Appendix B, with 372 steps per epoch; the model, AdamW, precision scaler, and random states are then restored, and the total schedule length is extended to 55800 steps while retaining the 3720-step warmup. At the first step after resumption, the learning rate is approximately 1.7818×10^{-5}, weight decay is approximately 0.2200, and the EMA coefficient is approximately 0.998; the learning rate at the end of 150 epochs is 10^{-6}. Results after 75 epochs therefore correspond to continued training under the extended schedule.

Table 27 retains all evaluation stages. Five-fold evaluation follows the original split of the pretraining images; class validation uses a fixed 80%/20% split grouped by image within the official training images, which were used in unlabeled pretraining. Official test evaluation uses target-token pooled and whole-image pooled features separately, both with a ridge-coefficient ratio of 10^{-4}.

I.2 Individual Training Trajectories and Learning Progress

To compare learning progress of the base-loss reference under the original and extended schedules, Table 27 lists all evaluation results from initialization to the endpoint for the two independent training runs.

Table 27: Learning curves for individual N0 training runs.

Run ID	Epoch	Step	Five-fold	Class validation	Target-token test	Whole-image pooled test
1	0	0	4.2895	4.2744	5.2403	7.1216
1	20	7440	13.5045	15.0021	15.2744	17.9108
1	40	14880	16.0102	17.7231	17.6821	20.3076
1	60	22320	16.5347	18.1297	18.6162	21.3475
1	75	27900	16.7057	19.1618	18.8945	21.6582
1	90	33480	17.4795	19.8394	19.4641	22.3529
1	100	37200	17.7179	20.0063	19.8330	22.6204
1	110	40920	17.7468	20.2773	20.2127	23.1705
1	120	44640	18.0673	20.5692	20.5126	23.4359
1	130	48360	18.0419	20.9341	20.5946	23.4726
1	140	52080	18.0203	20.9341	20.4910	23.6106
1	150	55800	18.0700	20.7360	20.5363	23.6473
2	0	0	4.0009	3.9095	5.4280	6.9598
2	20	7440	13.1370	14.2827	14.2238	16.7069
2	40	14880	15.2484	15.9821	16.7738	19.1038
2	60	22320	16.1907	17.3582	17.9151	20.2753
2	75	27900	16.3091	17.6918	18.2020	20.7413
2	90	33480	16.5714	18.6510	18.9722	21.6647
2	100	37200	16.8776	18.8803	19.4468	22.0314
2	110	40920	17.2858	18.7761	19.7187	22.5600
2	120	44640	17.4616	19.0575	20.1351	22.7822
2	130	48360	17.7334	19.2556	20.3896	23.2568
2	140	52080	17.8551	19.3807	20.4716	23.3819
2	150	55800	17.9342	19.2973	20.4824	23.4553

Note: Accuracy is reported in %; class validation and official test evaluation use their respective data splits. The first 75 epochs use the original schedule, followed by continued training under the extended schedule. N0 is the base-loss reference with no auxiliary loss or regularizer.

From 75 to 150 epochs, mean official test accuracy using target-token pooling increases by 1.9611 pp; from 130 to 150 epochs, it increases by only 0.0173 pp, indicating continued learning under the

extended schedule but slower gains at later stages. These two trajectories are used only to analyze the N0 learning process; the F/P interaction is still determined by comparisons across six independent training runs with the same budget.

The base prediction loss is sampled every 10 steps and averaged over the 2000 steps preceding each endpoint, using a left-open, right-closed interval. For the first and second independent training runs, it is 0.093617 and 0.086323, respectively, at 75 epochs and 0.095325 and 0.087748, respectively, at 150 epochs. Prediction loss increases slightly while class accuracy improves, indicating that prediction loss across stages and class readout must be evaluated separately during training in which the target encoder is updated by EMA.