CAN A MODEL LEARN FROM ARBITRARY PAIRS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive learning based on global embeddings has traditionally followed a simple rule (behind most supervised and self-supervised methods): semantically similar or identical samples should be pulled together, while distinct samples should be pushed apart. Despite its success, this learning principle does not use a pair of distinct classes as a positive. A snake and a cable can share an elongated outline, and a dog and a cow can share fur and a four-legged shape, but depicting that attribute is not trivial. This raises the question: *can a model learn from arbitrary class pairs without explicit guidance or attribute annotations?* The challenge is to pull a distinct pair together while preserving class discrimination. We address it with a discriminative subspace for every pair of classes, using a learning principle called SimLAP (**Sim**ple framework for Representation **L**earning from **A**rbitrary **P**airs). SimLAP introduces a feature filter that activates and deactivates pair-specific feature dimensions. An arbitrary class pair conditions a shared feature gate for all samples. The gate weights coordinates so that the contrastive loss imposes similarity for that pair. The positive is another view of the partner class, which is a random sample in the batch, so it may be the same class or a distinct class. With one shared gate, this objective matches a SupCon trained with the same recipe on ImageNet (75.8 vs. 76.1 linear probe) and CIFAR-100 (76.3 vs. 72.1 kNN). The gate is what makes that training succeed: without it, distinct and arbitrary positives collapse. Arbitrary pairs can provide useful representation learning signals for cross-class features, but their value depends on the semantic structure of the dataset. SimLAP makes such pairs trainable, providing a robust framework to exploit them.

1 INTRODUCTION

Humans exhibit remarkable visual perceptual abilities. We can group semantically similar objects and distinguish those that are different, while also identifying shared attributes across seemingly unrelated categories. For example, the colour of a green shirt and that of leaves share a common attribute despite belonging to entirely different semantic classes. Similarly, soaring birds resemble aircraft in shape, and many terrestrial mammals share structural characteristics such as four limbs, eyes, and a mouth despite belonging to distinct categories. Standard contrastive objectives do not directly impose this kind of cross-class similarity.

Conventional contrastive learning (CL) (Chen et al. (2020); He et al. (2020); Grill et al. (2020); Khosla et al. (2020); Wu & Wang (2020)) and self-supervised learning (SSL) (Oquab et al. (2023); Atito et al. (2021); Caron et al. (2020)) methods encode sample pairs as binary: two samples are either a positive pair (the same class, or two views of the same image) or a negative pair. Recent work has recognised that this assignment is too coarse. Hierarchical methods inject WordNet-style taxonomies or attribute annotations to encode structured inter-class similarity (Deng et al., 2014; Yan et al., 2015; Mohammed & Umaashankar, 2018; Ridnik et al., 2021). Closer to our motivation, Cabannes et al. (2023) show that both SL and SSL can be expressed through a similarity graph over samples, and that richer relational supervision improves representations—but they assume access to an oracle for those relationships. X-Sample Contrastive Loss (X-CLR) (Sobal et al., 2025) is directly about moving beyond the binary assignment: it replaces the one-hot positive/negative target with a continuous similarity graph via a pretrained semantic encoder so that every pair contributes a soft relation.

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.

These methods improve representation learning by expanding the set of positive pairs with hierarchical structure, attribute annotations, or similarity graphs. Explicitly specifying pairwise relations becomes increasingly difficult to scale as the number of classes grows, because the number of possible class pairs is quadratic. We raise a fundamental question: *can a model learn from arbitrary pairs of classes without this explicit relational guidance?* This raises two related questions: first, can arbitrary class pairs be used as positives without destroying class discrimination? Second, once such pairs are trainable, when do they provide useful signals for representation learning?

We answer the first question with SimLAP (Simple framework for Representation Learning from Arbitrary Pairs), which addresses a key challenge: *how to increase the similarity of pairs from semantically distinct classes while keeping the representation semantically discriminative*. Section 3.1 shows that two distinct classes can be similar in a subspace while the representation remains semantically discriminative. We therefore formulate representation learning from arbitrary pairs as imposing that similarity with a pair-conditioned gate, while class discrimination is preserved in the full space. This is achieved by a simple module, the *feature filter*. For a chosen class pair, the filter learns a gate vector with values in $[0, 1]$ that weights feature dimensions and thereby imposes similarity for that pair. SimLAP learns visual representations and subspaces for class pairs simultaneously by gradient descent. The gating mechanism is similar to conditional similarity networks (Veit et al., 2017) and makes SimLAP robust when learning from arbitrary pairs.

We answer the second question with experiments on ImageNet-1K and CIFAR-100, using class labels alone. ImageNet-1K has $1{,}000$ classes and $500{,}500$ class pairs; the feature filter learns a subspace for each pair by gradient descent (Section 5.2). Under the same training recipe, arbitrary-pair SimLAP is comparable to SupCon on ImageNet (75.8 versus 76.1 linear probe). On CIFAR-100, where many fine-grained classes share a coarse category, arbitrary pairs outperform identical pairs (76.3 versus 72.1 kNN). Arbitrary pairs therefore supply a useful signal when classes share structure, and stay close to identical-class training when many pairs are only weakly related.

In summary, our contributions are as follows: ❶ We formulate representation learning from arbitrary class pairs, where positive pairs are constructed without explicit attribute annotations or predefined class similarity. ❷ We introduce an adaptive filter that restricts cross-class attraction to a learned subspace, making distinct and arbitrary positives trainable while preserving discrimination in the global representation. ❸ We enhance SSL methods with arbitrary pairs to improve representation learning. ❹ We show that the usefulness of arbitrary pairs is dataset-dependent. On CIFAR-100, where many classes share coarse semantic structure, arbitrary and distinct pairs outperform identical pairs under SimLAP. On ImageNet, where the larger and more diverse class space makes unrelated random pairs more common, identical pairs remain stronger.

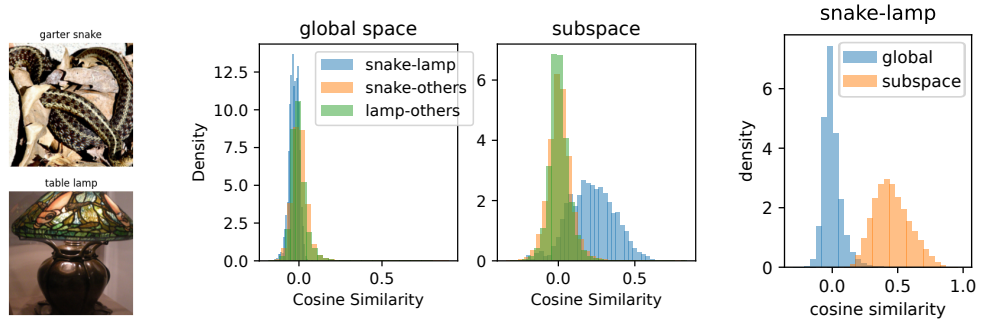
2 Related Work

Supervised Learning. Supervised classification encodes how classes relate through the target it assigns to each sample. Cross-entropy with a one-hot label (Krizhevsky et al., 2012) realises a binary class-similarity matrix: the target is one on the diagonal and zero for every other class, so each class is separated from the rest as if those classes were uniformly unrelated (Liu et al., 2016; Wang et al., 2016). Many classes are related, and the diagonal target does not capture this relationship. For example, the classes on ImageNet (Deng et al., 2009) follow a WordNet hierarchy (Fellbaum, 1998). Label-relation graphs score a prediction by semantic proximity (Deng et al., 2014), and hierarchical classifiers share representation along a taxonomy (Yan et al., 2015; Mohammed & Umaashankar, 2018; Ridnik et al., 2021). X-Sample Contrastive Loss (X-CLR) replaces the binary target with a continuous similarity read from a pretrained text encoder (Sobal et al., 2025). However, it is difficult to obtain the similarity matrix, and the text encoder introduces bias from pretraining. We use learnable class embeddings to capture the relationship between classes among the training data.

Contrastive Learning. Contrastive learning pulls positive pairs together and pushes negative pairs apart, usually with InfoNCE (Oord et al., 2018). There are mainly two conventions: Instance separation treats each image as one group–the positive is a different view of the anchor, and others are negative (Wu et al., 2018; Chen et al., 2020; He et al., 2020). Class separation fills the positive set with other samples of the anchor's label, which places the one-hot relation above inside a contrastive loss (Khosla et al., 2020). Cabannes et al. (2023) write both supervised and self-supervised training as optimisation over a sample similarity graph, and show that a few oracle relationships

Table 1: SimLAP differs from related methods in the positive set, the prior, and the condition.

Method	Positive set	Prior between classes	Conditional selection
SimCLR (Chen et al., 2020)	instance	none	none
SupCon (Khosla et al., 2020)	identical class	label	none
X-CLR (Sobal et al., 2025)	soft, all pairs	text encoder	none
Hierarchical learning (Ge et al., 2018)	identical class	taxonomy	none
CSN (Veit et al., 2017)	attribute match	named attributes	mask per attribute
Factorized spaces (Wang et al., 2023)	identical class	none	fixed split
SimLAP	arbitrary class	label	class pair



(a) Distinct class (b) Sign Filter (c) Learnable Filter

Figure 1: A distinct class pair is close in a subspace. **(a)** Garter snake and table lamp. **(b)** Sign filter on a supervised representation. **(c)** Learnable filter of SimLAP, which raises the snake–lamp similarity to about 0.48 (eq. (8)). The two classes stay dissimilar in the full space.

improve the representation. Structure shared by different classes becomes a training signal when those classes are themselves admitted as positives. However, applying contrastive learning directly in the full space imposes an incorrect semantic relation (for example, pulling a lamp toward a snake rather than toward a chair) and leads to representation collapse. Our method addresses this issue by learning a pair-specific subspace for each class pair.

Conditional and Subspace Methods. A subspace is a way to train several similarity objectives on one embedding. Each objective acts on its own coordinates, so the objectives need not agree in the full space. Factorized spaces split the embedding into fixed slices and apply a triplet loss inside each slice (Sanakoyeu et al., 2019; Wang et al., 2023). Conditional Similarity Networks (CSN) (Veit et al., 2017) learn the split. They keep one embedding and one mask for each named similarity, such as colour or category, and apply a triplet loss inside the masked coordinates. The learning rule matches SimLAP: the mask is a coordinate gate, and the loss pulls a pair together only on the coordinates that the gate keeps. CSN's positive must match the anchor on a named attribute. SimLAP does not: the partner class is randomly sampled from the batch, and no attribute annotation is required to build $\mathbb{P}$.

Table 1 summarizes the differences between SimLAP and related methods.

3 Learning from Arbitrary Pairs of Classes

We introduce a novel learning paradigm that diverges from the conventional objective of maximising intra-group (i.e., class or instance) similarity while minimising inter-group similarity. Instead, we optimise similarity between *arbitrary pairs of classes* under discriminative subspaces. An arbitrary pair is a class pair (y_i, y_j). We obtain it from two randomly selected samples $(\boldsymbol{x}_i, \boldsymbol{x}_j)$, with no requirement that $y_i = y_j$ and without assuming any predefined semantic relationship or inter-class structure. The detailed implementation is in Section C.1 and Algorithm 1.

3.1 Motivation for Learning in Subspaces

A key challenge is *how to pull even distinct class pairs together without diminishing semantic discrimination in the representation.* In the absence of explicit relational guidance, we initially regard every class pair as potentially positive. However, symmetrically attracting all pairs collapses the representation space into conflicting regions. Intuitively, a pair-conditioned gate imposes similarity on a subset of coordinates, so the two classes need not be attracted in the full space.

Before formulating our approach, we investigate whether current learning algorithms already exhibit similarity between arbitrary pairs in subspaces. We consider a hard example: two distinct classes with no apparent common features, 'garter snake' and 'table lamp' (Figure 1a). Using a pretrained ResNet-50 trained with supervised learning on IN1K from timm (Wightman, 2019), we extract features for all validation samples of these two classes. We compare the similarity of these class pairs in the full space and in the filtered subspace. A sign filter, which keeps dimensions of the same sign, raises their similarity (Figure 1b; calculation in Section C.2). SimLAP uses a learnable filter instead (Figure 1c). The two distinct classes are nearly orthogonal in the global space, yet close in the selected subspace. **Observation:** *For high-dimensional representations, we can find a subspace for an arbitrary class pair such that their samples are close to each other and far away from those of other classes in that subspace.*

3.2 Contrastive Learning

An encoder f_V and a projector f_P map an image to a feature $\boldsymbol{z} = f_P(f_V(\boldsymbol{x}))$ (Chen et al., 2020; Jing et al., 2022). As in standard contrastive learning, each image is encoded as two augmented views, and both views sit in the mini-batch. The anchor view itself is excluded from $\mathbb{P}$ and from $\mathbb{N}$. Let $\langle u, v\rangle = \frac{u \cdot v}{\|u\|\|v\|}$ denote cosine similarity. For an anchor $\boldsymbol{z}_i$, with positives $\boldsymbol{z}^+ \in \mathbb{P}$ and negatives $\boldsymbol{z}^- \in \mathbb{N}$, InfoNCE (Oord et al., 2018) is

$$\mathcal{L}(\boldsymbol{z}_i) = -\log \frac{\exp(\langle \boldsymbol{z}_i, \boldsymbol{z}^+\rangle/\tau)}{\exp(\langle \boldsymbol{z}_i, \boldsymbol{z}^+\rangle/\tau) + \sum_{\boldsymbol{z}^- \in \mathbb{N}} \exp(\langle \boldsymbol{z}_i, \boldsymbol{z}^-\rangle/\tau)}. \tag{1}$$

SimCLR draws $\mathbb{P}$ from another view of the anchor (Chen et al., 2020); SupCon draws it from the same class (Khosla et al., 2020).

3.3 Learning from Arbitrary Pairs

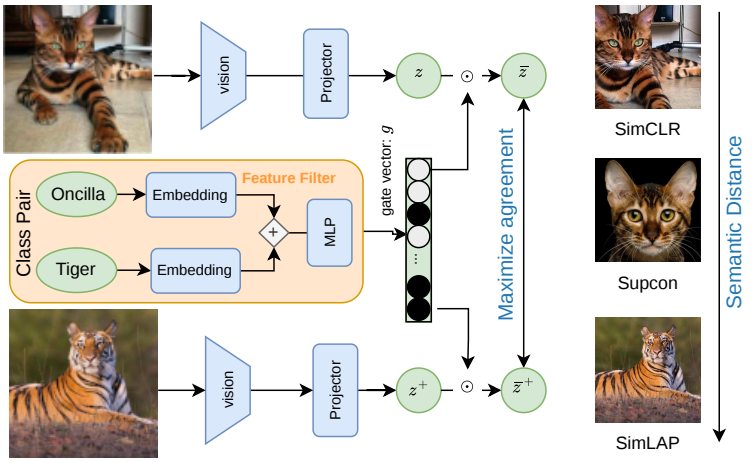


Figure 2: (a) The architecture of SimLAP. A feature gate imposes similarity for a class pair. (b) Comparison of pair construction in SimCLR, SupCon, and SimLAP.

SimLAP performs contrastive learning on filtered features, with one gate vector per class pair (Figure 2). Let y_j be the partner class. $\mathbb{P}$ is every other mini-batch sample of class y_j, excluding the anchor. $\mathbb{N}$ is every remaining sample. Under arbitrary pairing, y_j is a random label in the batch, so it may equal the anchor class. A random permutation π of the mini-batch $\{(\boldsymbol{x}_k, y_k)\}_{k=1}^{N}$ pairs $\boldsymbol{x}_i$ with $\boldsymbol{x}_{\pi(i)}$ and induces the class pair y_i-y_j, where $y_j = y_{\pi(i)}$ (Section C.1). The feature filter maps the pair to a gate $g(y_i, y_j)$ with values in $[0, 1]$ (eq. (3)). The same gate multiplies all samples, $\bar{\boldsymbol{z}}_i = g(y_i, y_j) \odot \boldsymbol{z}_i$, and likewise for each positive and each negative. The anchor loss averages one term per member of $\mathbb{P}$. The denominator of each term is that positive together with $\mathbb{N}$:

$$\mathcal{L}(y_i, y_j) = -\frac{1}{|\mathbb{P}|} \sum_{\bar{\boldsymbol{z}}^+ \in \mathbb{P}} \log \frac{\exp(\langle \bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}^+ \rangle / \tau)}{\exp(\langle \bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}^+ \rangle / \tau) + \sum_{\bar{\boldsymbol{z}}^- \in \mathbb{N}} \exp(\langle \bar{\boldsymbol{z}}_i, \bar{\boldsymbol{z}}^- \rangle / \tau)}. \tag{2}$$

Every comparison stays in the subspace of y_i-y_j, so the anchor is separated from other classes along the dimensions the gate keeps. Section A.1 sketches this mechanism: the gate gradient can up-weight coordinates that support the chosen pair and down-weight coordinates that do not.

Table 2: **Top-1 accuracy on ImageNet.** All models are trained for 200 epochs.

Model	Linear	Finetune
CE	77.10	73.70
SimCLR	65.59	76.0
SupCon	**76.08**	77.56
SimLAP	75.81	**77.72**

Table 3: **Linear probe accuracy** on ImageNet. All models are trained for 100 epochs.

Loss combination	Top-1	Top-5
SimCLR	63.94	85.82
SimCLR + SupCon	73.70	91.52
SimCLR + SupCon + SimLAP	**73.89**	**91.65**

Table 4: **Transfer Learning with kNN evaluation** (K=10). SimCLR, SupCon, and SimLAP are trained for 200 epochs. X-CLR and X-SimLAP (100 epochs) use Sentence Transformer (Reimers & Gurevych, 2019) to encode class embeddings.

Model (epochs)	Aircraft	CF10	Cars	DTD	FLW	Food	Pets	STL	AVG
SimCLR (200)	20.58	83.10	14.24	66.60	73.12	51.28	72.28	93.06	59.28
SupCon (200)	**45.42**	**90.79**	**44.89**	67.87	87.85	**63.40**	89.04	**96.97**	**73.28**
SimLAP (200)	45.27	90.09	43.74	**68.03**	**88.47**	63.22	**89.78**	96.91	73.19
X-CLR (100)	39.96	89.12	39.76	**66.01**	85.61	62.16	**89.59**	96.80	71.13
X-SimLAP (100)	**45.06**	**89.92**	**43.88**	65.85	**87.22**	**62.29**	89.56	**97.15**	**72.62**

Feature Filter. The pair descriptor is the mean of the two label vectors, $\mu(y_i, y_j) = (f_l(y_i) + f_l(y_j))/2$. The feature filter maps a class pair to its gate. It consists of two parts, a label embedding f_l and an MLP f_g. The label embeddings are randomly initialised, carrying no class semantics, to convert discrete labels to continuous vectors. The embeddings are shared across pairs and trained with the encoder. Figure 9 shows the super-synset structure they learn from the visual encoder. The MLP maps the pair descriptor to a gate that weights dimensions for the pair. The weights impose similarity for the pair. The calculation is

$$g(y_i, y_j) = \sigma(f_g(\mu(y_i, y_j))), \tag{3}$$

where $f_g(\cdot)$ denotes the MLP. Empirically, the learned gates converge to roughly binary values (activated or suppressed), with Gaussian-like activation patterns across class pairs (Figure 6), so we do not regularise the gate toward sparsity.

4 Experiments

4.1 Experimental Setting

We train SimLAP on ImageNet (Deng et al., 2009) with a ResNet-50 (He et al., 2016) backbone for 200 epochs at a global batch of 4096. The training recipe is LARS, a base learning rate of

$0.4 \times \text{batch}/256$ (peak 6.4) with cosine decay, weight decay 10^{-6}, a 12-epoch linear warmup, a two-layer MLP projector to 256 dimensions, temperature 0.1, and colour jitter. Each anchor uses one subspace, the gate of its class pair. SimCLR and SupCon have the same projector, except SimLAP ends in LayerNorm, which the gate requires. Architecture, optimizer, and augmentation details are in Section C.3 and Table 12. SimCLR, SupCon, and SimLAP all use this 200-epoch budget, so the comparison sees the same amount of data. This section tests whether arbitrary-pair training stays competitive with same-class training.

Compared Baselines. We compare SimLAP with widely adopted methods as reference points rather than as targets to outperform, including supervised learning with cross-entropy and one-hot vectors (CE) (He et al., 2016), supervised contrastive learning (SupCon) (Khosla et al., 2020), and instance contrastive learning (SimCLR) (Chen et al., 2020). Considering the different training settings for these methods, we use the same training recipe as SimLAP to reproduce them for a fair comparison. These reference methods are chosen to evaluate two aspects of the framework: *(i) whether the gate keeps training stable for arbitrary and distinct class pairs, and (ii) whether that training stays competitive with conventional SL and SSL.*

4.2 Main Results

Classification Accuracy. We use two protocols to evaluate the learned representations: a linear probe of the frozen embeddings, and fine-tuning to expose semantics stored in the network. SimLAP serves as a pretraining objective; the head does not impose a global class separation. Table 2 shows that SimLAP learns class discrimination comparable to the supervised contrastive baseline. Fine-tuning raises SimLAP from 75.81% to 77.72% and SupCon from 76.08% to 77.56%. Table 3 shows that adding SimLAP to SimCLR+SupCon moves the linear probe from 73.70% to 73.89%.

Transfer Learning. We adopt the kNN protocol to probe transferable representations without any additional training. We compare SimLAP with the two typical CL methods: SimCLR and SupCon. Table 4 shows that SimLAP is comparable to SupCon (-0.09%) and surpasses the instance-contrastive baseline ($+13.9\%$).

Class Prior. Class similarity graphs have been proven to improve representation learning. Our framework is flexible to use a sentence transformer (Reimers & Gurevych, 2019) which encodes class names as label embeddings. Table 4 shows a comparison with X-CLR (Sobal et al., 2025), and our version (X-SimLAP) achieves $+1.49\%$ improvement.

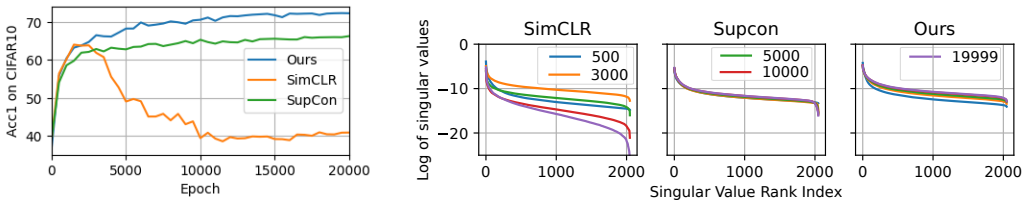
Video object segmentation. We use DAVIS-2017 video object segmentation (Pont-Tuset et al., 2017) to evaluate local features for dense tasks, following DINO (Caron et al., 2021). Frozen ResNet-50 features propagate the first-frame masks by nearest-neighbor matching between consecutive frames, with no dense head trained on top. We report the mean over 61 objects on the 30-video validation split at stride 32: region similarity $\mathcal{J}_m$, contour accuracy $\mathcal{F}_m$, and their average $(\mathcal{J}\&\mathcal{F})_m$ (Table 5). SimCLR is the strongest ($(\mathcal{J}\&\mathcal{F})_m$ 39.6). SimLAP and SupCon are very close (38.0 vs. 38.4), and all three contrastive models exceed CE. Learning from arbitrary pairs therefore keeps the spatial detail needed for dense correspondence.

Table 5: **Video object segmentation** on DAVIS-2017 val. Frozen ResNet-50 features.

	$\mathcal{J}_m$	$\mathcal{F}_m$	$(\mathcal{J}\&\mathcal{F})_m$
CE	41.8	28.2	35.0
SimCLR	**44.9**	**34.2**	**39.6**
SupCon	42.9	33.9	38.4
SimLAP	42.3	33.7	38.0

Table 6: CIFAR-10 kNN vs. number of optimized subspaces per step.

Num	GFlops	Time	act.	CF10
1	1,063	0.30	63	85.14
5	1,064	0.32	53	85.96
20	1,067	0.39	28	**86.21**



(a) Performance during training

(b) Singular value spectrum of the embedding space

Figure 3: **Learning stability for long-training (20,000 epochs)**. Lower singular values suggest that representations are concentrating information in fewer dimensions. Models trained on IN1P (10 dogs) and evaluated with kNN on CIFAR-10 show that SimLAP is more stable than SimCLR.

Table 7: CIFAR-100 (kNN) for **Pairs and gates**.

Gate	Identical	Distinct	Arbitrary
open	72.18	4.77	6.43
fixed	**72.23**	3.93	6.46
asym	5.78	5.01	4.35
sym	72.13	**76.15**	**76.26**

Table 8: Learning from **different positives** on CIFAR-10 (kNN, k=10), with no gate: a mixup sample or the mean representations.

Positive z^+	Identical	Arbitrary
Mixup samples	91.91	26.5
Mean representations	90.34	27.33

5 Model Analysis

5.1 Ablation Study

Pair Types and Gates. We ablate pair type and the gate on CIFAR-100, using its 100 fine-grained classes (Krizhevsky et al., 2009). Every run trains a ResNet-50 from scratch for 100 epochs with batch size 128, and we report kNN top-1 (k=10). The positive set is unchanged across gates: other-view samples of the partner class are positives, and the anchor class is excluded unless it is the partner. Only the partner and the gate change.

An identical pair sets the partner label equal to the anchor label. A distinct pair draws the partner from a different class. An arbitrary pair sets the partner by a random permutation of the batch labels, so the partner includes identical and distinct pairs. This last construction is the default of SimLAP. 'open' compares features in the full space, removing the gate. 'fixed' draws one mask $m \in \{0,1\}^D$ at the start and freezes it. The same mask multiplies the anchor and every candidate. 'asym' uses a learnable class gate $g(y)$. Each sample keeps its own gate during the comparison, $\langle g(y_i) \odot z(x_i),\ g(y_k) \odot z(x_k) \rangle$, so the contrast does not share one gate across the anchor and the candidates. The InfoNCE positives still follow the pair type above, including the random partner labels. 'sym' is the default of SimLAP: one learnable pair gate $g(y_i, y_j)$ multiplies the anchor and every candidate in that anchor's loss (eq. (2)).

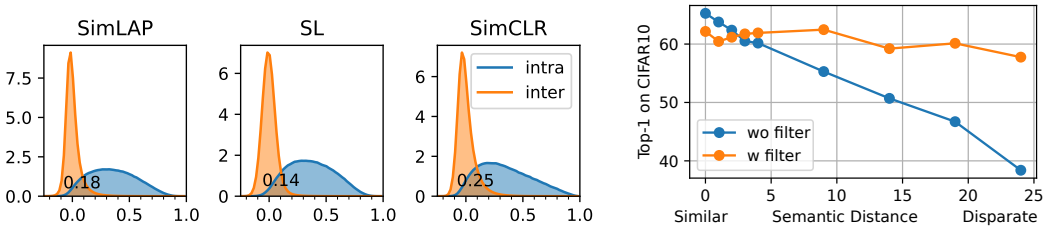
Table 7 crosses these pairs with four gates. The results fall into two regimes: (1) success, with accuracy above 60%; (2) failure, with accuracy below 10%. Without the gate, CL only works for identical pairs. The asymmetric gate, applying different activation to anchor and candidate, fails on all pair types. Our method, the symmetric gate, is robust to all pair types. We can see advantage of arbitrary pairs over identical pairs in CIFAR-100, because many fine-grained classes

Table 9: kNN (k=10) with ViT-Tiny for $\frac{1}{1+\beta}\mathcal{L}_{\text{MoCo}} + \frac{\beta}{1+\beta}\mathcal{L}_{\text{SimLAP}}$.

beta	CIFAR10	DTD	FLW
0	82.90	63.35	83.10
1	**90.33**	**64.31**	**88.18**
100	87.72	58.51	85.23

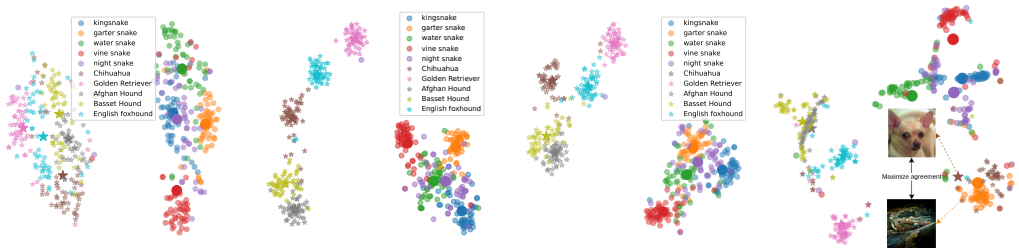
Table 10: Top-1 accuracy on ImageNet for finetuning pretrained models with CE and SimLAP with ViT-Base.

Pretrained	CE	CE-CE	SimLAP-CE
MAE	83.6	83.1 (−0.5)	84.0 (+0.4)
DINOv2	86.2	85.7 (−0.5)	86.4 (+0.2)
MoCo	83.2	82.8 (−0.4)	83.8 (+0.6)



(a) **Inter-/intra-class similarity** on IN1K. (b) **Filter vs. semantic distance**.

Figure 4: (a) Overlap of inter- and intra-class similarity (number in the middle); see eq. (8). (b) CIFAR-10 top-1 vs. CLIP-ranked semantic distance of the positive set. Without the filter, distinct pairs degrade the representation.



(a) SimCLR (b) SupCon (c) SimLAP (d) SimLAP (subspace)

Figure 5: t**-SNE** of dog (star) and snake (dot) embeddings. (d) shows SimLAP aligning Chihuahua and garter snake in a subspace, while (c) keeps the classes separated.

are semantically related belonging to the same coarse-grained class. We also run the experiment on ImageNet with different class pairs in Table 11. On ImageNet the order reverses: identical pairs score 69.80 average kNN, distinct 61.39, and arbitrary 60.57. Arbitrary pairs remain trainable; they are not the stronger sampling policy due to large number of diverse classes.

Cross-class Features. We also ablate two methods to fuse cross-class features for positives: mixup samples and mean representations. Table 8 shows the performance of different positive sets. The models were trained and evaluated on CIFAR-10. The low performance on arbitrary pairs shows that models cannot learn meaningful representations from arbitrary pairs without gates.

5.2 Why SimLAP Works

Finding 1: *Subspace training preserves global class separation.* The objective does not explicitly impose a single global class-separation geometry; instead, class separation emerges from multiple pair-conditioned contrastive objectives. Backbone embeddings of SimLAP (excluding the projector and filter) remain class-discriminative. On IN1K, inter-/intra-class similarity overlap for SimLAP is higher than for SL—expected, since SL separates classes directly—yet lower than for SimCLR in Figure 4a, showing that subspace optimization still induces global separation. t-SNE (Van der Maaten & Hinton, 2008) of 10 IN1K validation classes in Figure 5 confirms clear

Table 11: **Top-1 accuracy on ImageNet for different class pairs**. All models are trained for 100 epochs. Use kNN evaluation with k=10.

Pair	Aircraft	CF10	Cars	DTD	Flowers	Food	Pets	STL	AVG
Arbitrary	25.77	82.50	26.65	56.54	62.86	44.99	88.20	97.11	60.57
Distinct	24.09	84.06	25.03	59.84	68.37	45.45	87.49	96.86	61.39
Identical	**36.69**	**87.69**	**38.86**	**66.01**	**83.62**	**58.10**	**90.02**	**97.44**	**69.80**

global structure comparable to SupCon, while a learned subspace aligns distinct pairs such as Chihuahua and garter snake.

Finding 2: *SimLAP simultaneously optimises representations from visual data and subspaces for class pairs.* The encoder and the gate are trained together. The gate imposes similarity for each class pair. By default, each anchor uses one subspace, the gate of its class pair. Specifically, we first compute global features by forwarding images once, then generate multiple subspaces. In Table 6, we report FLOPs and time per training step, 'act.' as the number of active dimensions, and 'CF10' as top-1 accuracy on CIFAR-10. Increasing the number of optimised subspaces increases training time but improves representations and yields smaller, more discriminative subspaces.

Finding 3: *The filter makes learning robust to distinct pairs.* We design a controlled experiment to increase the semantic distance in the positive set and see whether the gate still lets training succeed. We run the experiment on IN25, a dataset of 25 classes with well-defined semantic relationships (details in Section C.3). Each model was trained for 1000 epochs. As shown in Figure 4b, the performance of the model without the filter degrades as semantic distance increases, because distinct pairs impose an incorrect semantic relationship between classes. In contrast, the model with the filter can learn transferable representations from distinct pairs.

Finding 4: *SimLAP shows greater learning stability in our long-training IN1P experiment.* We train each model for 20,000 epochs on IN1P to examine the learning stability (cf. Section C.3). In Figure 3a, we compare kNN top-1 accuracy on CIFAR-10; Figure 3b shows the corresponding singular-value spectra. SimCLR exhibits signs of dimensional collapse (Jing et al., 2022) after 3,000 epochs: declining transfer accuracy and shrinking singular values. These results demonstrate the learning stability in this long-training IN1P setting for SimLAP.

6 Application

Joint Loss. The simplest way to combine SimLAP and MoCo v3 is to train with a joint loss $\frac{1}{1+\beta}\mathcal{L}_{\text{MoCo}} + \frac{\beta}{1+\beta}\mathcal{L}_{\text{SimLAP}}$ cf. Section C.3. We follow the MoCo v3 setting (Chen et al., 2021) for ViT-Tiny. The mixture reduces to MoCo v3 at $\beta = 0$. At $\beta = 100$ the SimLAP weight is $100/101$, so that run is approximately SimLAP . As shown in Table 9, kNN (k=10) evaluation of ViT-Tiny is highest at $\beta = 1$, with gains of $+7.43$ (CIFAR-10), $+0.96$ (DTD), and $+5.08$ (Flowers) over MoCo v3.

Mid-training. SimLAP is a 100-epoch bridge between a pretrained ViT-Base and 100 epochs of finetuning with cross-entropy on ImageNet cf. Section C.3. Table 10 shows that continuing CE for another 100 epochs instead lowers top-1 by 0.4–0.5. The 200-epoch training with SimLAP-first then CE improves top-1 by MAE $+0.4$, DINOv2 $+0.2$, and MoCo $+0.6$ relative to the 100-epoch CE baseline.

7 Discussion

Our study shows that representation learning from arbitrary class pairs is feasible in conditional subspaces when the anchor and candidates share a pair-conditioned gate. On ImageNet, arbitrary-pair SimLAP achieves 75.81% linear-probe accuracy, compared with 76.08% for SupCon trained for the same settings. On CIFAR-100, arbitrary pairs outperform identical pairs within SimLAP (76.26% versus 72.13% kNN), revealing the value of arbitrary pairs in datasets with many semantically related classes. Together, SimLAP addresses the feasibility of learning from arbitrary pairs, while the benefit of cross-class positives depends on the distribution of informative pair relations in the dataset. We therefore treat arbitrary pairing as a demonstration of an additional training signal, rather than a general-purpose sampling policy. Combining this objective with supervised or self-supervised training is promising.

Limitation & Future Work. We use random sampling to select the class pairs, which may not fully explore the available space. Moreover, the carried information among a sample pair is distinct from the sample itself, so learning on low-value pairs can be inefficient. How to mine informative pairs remains an open challenge. Our method requires class labels to condition the feature filter, so it is limited in supervised settings; extending it with pseudo-labels is left for future work.

AI USE STATEMENT

In this work, we have not used generative AI tools to generate synthetic datasets, design the experimental protocol, clean or reformat datasets, or interpret results. We have used generative AI tools to draft the findings from experiments and improve the writing by correcting grammar and correcting the claims, search related literature, and assist with software implementation and experiment launch. All AI-assisted text and code were reviewed and verified by the authors. The gradient analysis is drafted by us and later corrected by Grok 4.6. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Sara Atito, Muhammad Awais, and Josef Kittler. Sit: Self-supervised vision transformer. *arXiv preprint arXiv:2104.03602*, 2021.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

Vivien Cabannes, Leon Bottou, Yann Lecun, and Randall Balestriero. Active self-supervised learning: A few low-cost relationships are all you need. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 16274–16283, October 2023.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/70feb62b69f16e0238f741fab228fec2-Abstract.html.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9640–9649, 2021.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Jiequan Cui, Zhisheng Zhong, Zhuotao Tian, Shu Liu, Bei Yu, and Jiaya Jia. Generalized parametric contrastive learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

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

Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *European conference on computer vision*, pp. 48–64. Springer, 2014.

Christiane Fellbaum. *WordNet: An electronic lexical database*. MIT press, 1998.

Weifeng Ge, Weilin Huang, Dengke Dong, and Matthew R. Scott. Deep metric learning with hierarchical triplet loss. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 269–285, 2018.

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

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

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=YevsQ05DEN7.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

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

Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. *arXiv preprint arXiv:1612.02295*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Abdul Arfat Mohammed and Venkatesh Umaashankar. Effectiveness of hierarchical softmax in large scale classification tasks. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1090–1094. IEEE, 2018.

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

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

Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *arXiv:1704.00675*, 2017.

Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5706–5715, 2018.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pp. 3980–3990. Association for Computational Linguistics, 2019. doi: 10.18653/V1/D19-1410. URL https://doi.org/10.18653/v1/D19-1410.

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

Artsiom Sanakoyeu, Vadim Tschernezki, Uta Büchler, and Björn Ommer. Divide and conquer the embedding space for metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 471–480, 2019.

Vlad Sobal, Mark Ibrahim, Randall Balestriero, Vivien Cabannes, Diane Bouchacourt, Pietro Astolfi, Kyunghyun Cho, and Yann LeCun. $\mathbb{X}$-sample contrastive loss: Improving contrastive learning with sample similarity graphs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=c1Ng0f8ivn.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Andreas Veit, Serge Belongie, and Theofanis Karaletsos. Conditional similarity networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 830–838, 2017.

Chengkun Wang, Wenzhao Zheng, Junlong Li, Jie Zhou, and Jiwen Lu. Deep factorized metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Lin Wang, Bo Yang, Yuehui Chen, Xiaoqian Zhang, and Jeff Orchard. Improving neural-network classifiers using nearest neighbor partitioning. *IEEE transactions on neural networks and learning systems*, 28(10):2255–2267, 2016.

Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Jiantao Wu and Lin Wang. Arcgrad: Angular gradient margin loss for classification. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2020. doi: 10.1109/IJCNN48605.2020.9207251.

Jiantao Wu, Shentong Mo, Sara Atito, Zhenhua Feng, Josef Kittler, and Muhammad Awais. Dailymae: Towards pretraining masked autoencoders in one day. *arXiv preprint arXiv:2404.00509*, 2024.

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

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 2740–2748, 2015.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

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.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2018. URL https://arxiv.org/abs/1710.09412.

Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 13001–13008, 2020.

Appendix

A Analysis of SimLAP

A.1 Gradient Analysis

This behavior can also be explained from the point of view of gradients. Consider a simplified situation, where C samples are drawn from C distinct classes, denoted (x_i, y_i). The pairwise similarity score between samples i and j is defined as $s(i,j) = \langle \mathbf{g} \odot z_i, \mathbf{g} \odot z_j \rangle / \tau$, where τ is the temperature and $\mathbf{g} = g(y_i, \bar{y}_i) \in [0,1]^D$ is the gate vector of the subspace for the class pair $(y_i, \bar{y}_i)$, with $\bar{y}$ the shuffled labels (Algorithm 1). Let j denote the index of the positive sample paired with the anchor i. The objective L is the symmetrized InfoNCE loss for the positive pair (i,j):

$$L = -\log\left(\frac{\exp(s(i,j))}{\sum_{k\neq i}\exp(s(i,k))}\right) - \log\left(\frac{\exp(s(j,i))}{\sum_{k\neq j}\exp(s(j,k))}\right). \tag{4}$$

Because $s(i,k)$ is a cosine, both gated norms depend on $\mathbf{g}$. For a candidate k, write

$$A_{ik} = \sum_r (g^r)^2 (z_i^r)^2, \quad B_{ik} = \sum_r (g^r)^2 (z_k^r)^2, \quad c_{ik} = \frac{\sum_r (g^r)^2 z_i^r z_k^r}{\sqrt{A_{ik} B_{ik}}}, \tag{5}$$

so $s(i,k) = c_{ik}/\tau$ whenever both norms are nonzero. Differentiating the numerator and the two norms gives

$$\frac{\partial s(i,k)}{\partial g^d} = \frac{1}{\tau}\left[\frac{2 g^d z_i^d z_k^d}{\sqrt{A_{ik} B_{ik}}} - c_{ik}\, g^d \left(\frac{(z_i^d)^2}{A_{ik}} + \frac{(z_k^d)^2}{B_{ik}}\right)\right]. \tag{6}$$

The second term is the contribution of the norms. Dropping it yields the gradient of an unnormalized gated dot product, not of the cosine used in the loss. The loss gradient is the softmax-weighted sum of these score derivatives. With $p(x_k \mid i) = \exp(s(i,k)) / \sum_{m\neq i}\exp(s(i,m))$,

$$\begin{aligned}\frac{\partial L}{\partial g^d} = &\sum_{k\neq i,j} p(x_k \mid i)\,\frac{\partial s(i,k)}{\partial g^d} - \big(1 - p(x_j \mid i)\big)\,\frac{\partial s(i,j)}{\partial g^d} \\ &+ \sum_{k\neq i,j} p(x_k \mid j)\,\frac{\partial s(j,k)}{\partial g^d} - \big(1 - p(x_i \mid j)\big)\,\frac{\partial s(j,i)}{\partial g^d}.\end{aligned} \tag{7}$$

Gradient descent increases g^d when $\partial L / \partial g^d < 0$. The coordinate product $z_i^d z_k^d$ does not by itself set that sign: the norm term in eq. (6) can oppose it. This partial derivative is also not a free update of each gate coordinate. The training gate is $\sigma(f_g(\mu(y_i, y_j)))$, so the shared MLP and the sigmoid sit between $\mathbf{g}$ and the parameters. The two-anchor loss above is only an illustration; a random permutation of the mini-batch need not form disjoint pairs. Near-binary gates are an empirical observation (Figure 6), not a consequence of eq. (7), and the local gradient does not prove that optimization is stable.

B A Case Study

We visualize the gate values of SimLAP in Figure 6. Specifically, we pass 1K classes to the filter to generate 1000 vectors with 256-D gate values, which indicate the activation of the corresponding dimensions for each class. Each light point on the heatmap represents the activation status of a specific dimension for the corresponding class. From the figure, we can observe that the gates exhibit roughly binary values (either activated or not), and the activation vectors for different classes are distinct. This indicates that each class creates a unique subspace within the feature space. These experimental observations support our hypothesis that SimLAP can effectively learn to create class-level subspaces.

Considering the large number and diversity of classes in IN1K, we use a case study to demonstrate how SimLAP interprets the similarity between classes. Specifically, we select 17 classes belonging to the super synset 'snake.n.01', 16 classes belonging to the super synset 'wading bird.n.01', and

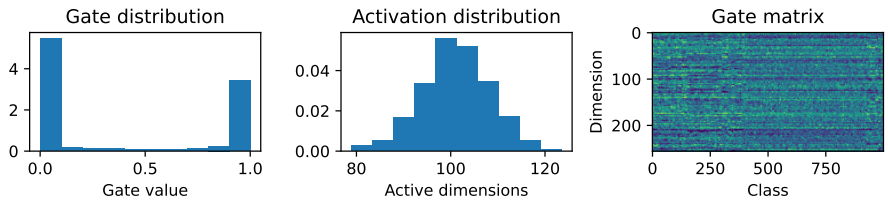


Figure 6: **Gate visualization**. Active dimensions are identified by the sum of gate values for each class, which defines the size of the subspace. The gate matrix shows the activation of one dimension for a class. The filter selects different dimensions for each class, so the subspaces are class-specific.

21 classes belonging to the super synset 'furniture.n.01' from the IN1K validation set. Each class contains 50 images. This case study helps us understand that SimLAP promotes cross-class features as well as class-level features.

Semantic Hierarchy. Synsets are sets of cognitive synonyms, such as 'hunting_dog.n.1', defined in WordNet (Fellbaum, 1998). Synsets are inter-linked by a sparse network of typed conceptual relations (hypernymy, hyponymy, meronymy, etc.). Each synset carries a concise definition ("gloss") and, wherever useful, one or two illustrative sentences. ImageNet is organised according to WordNet, so we know which classes belong to a super-synset, for example the 21 classes under 'furniture.n.01'.

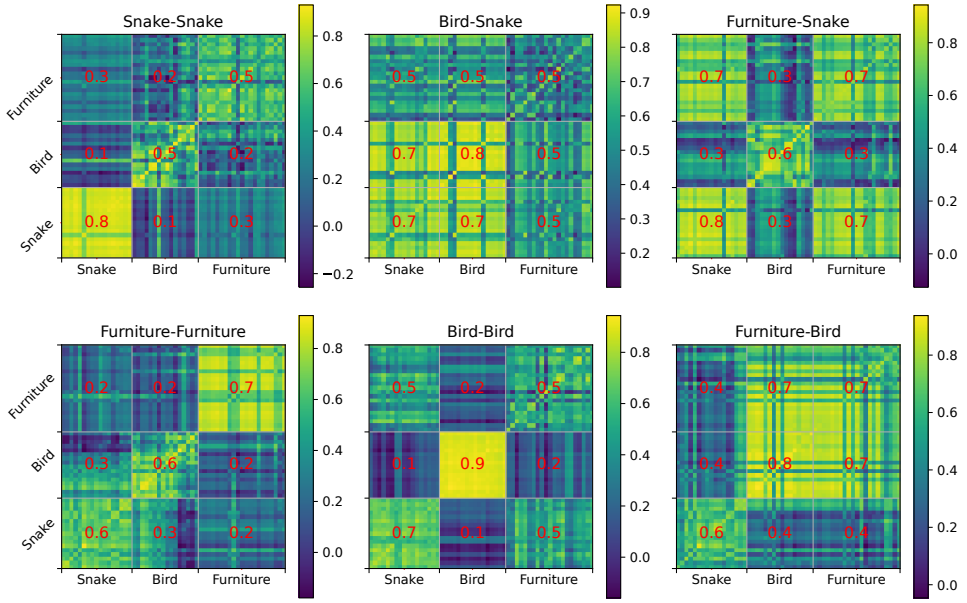


Figure 7: **Class similarity in subspaces.** The gap between intra and inter coarse-class similarity is widened in their corresponding subspaces, highlighting SimLAP's effectiveness in creating discriminative subspaces.

Similarity after gating. For subclasses inside one coarse category, global features are already similar. We then keep the ten coordinates with the highest gate values in that category (Figure 8) and recompute class similarity on those coordinates. Figure 7 visualizes the class similarity in the subspaces computed for three coarse classes: Snake, Bird, and Furniture. Compared to the class similarity in the global space, the discriminative information is strengthened for the corresponding subspace. Specifically, the gap between intra and inter coarse-class similarity is magnified to 0.5 for Snake in the Snake-Snake Subspace, 0.7 for Bird in the Bird-Bird Subspace, and 0.5 for Furniture in the Furniture-Furniture Subspace. The features in the subspaces must contain distinct information about the class pair. These features also contain visual information about other classes such that

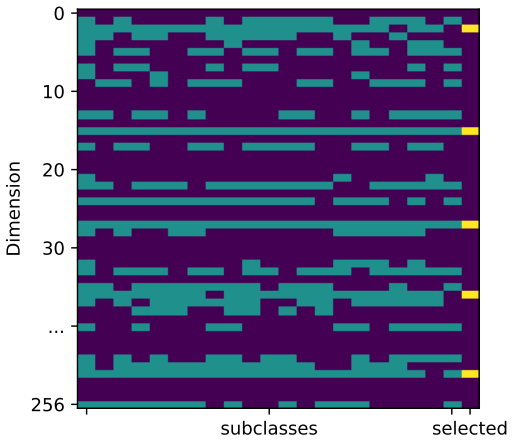
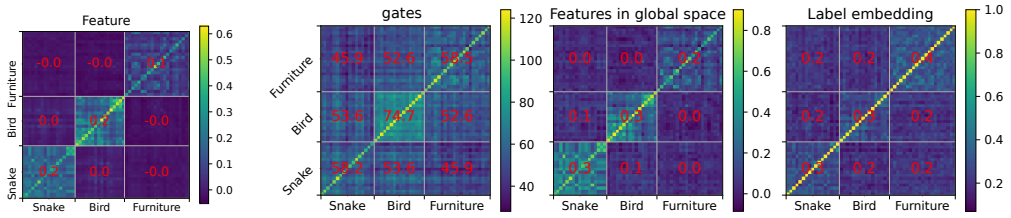


Figure 8: Gate values for garter snake and table lamp. Yellow marks coordinates with the highest average gate value.



(a) SupCon (b) SimLAP

Figure 9: **Class similarity measured by class-wise gates, global features, and label vectors.** Red numbers denote the average similarity within each super synset region (snake, wading bird, furniture). Higher class similarity of super synset indicates that feature filter in SimLAP learns the relationships between classes from representations.

these classes are still discriminative in the unrelated subspaces. The gate changes the measured similarity, including for Furniture–Snake, where global similarity is low.

Class Similarity in Global Space. We investigate three similarity measurements to evaluate the ability to identify the relationships between classes: 1) *Dot product of class-wise gates*: We calculate the gates for each class and sharpen the values to 1 if above 0.5 and to 0 otherwise ($g(y, y)$). This measure indicates how similar the activated dimensions are between classes, reflecting the structural similarity of their subspaces. 2) *Features in the global space*: For each class pair, we use the global features to calculate the cosine similarity between all sample pairs coming from each class pair. We then use the average to denote the similarity between classes. This represents the overall visual similarity in the original feature space. 3) *Class similarity in label embedding*: We obtain the class vectors through the label embedding in the feature filter, and then calculate their cosine similarity. This measure reflects the learned semantic relationships between class labels. Figure 9 compares these three similarity measurements. Gates and label embeddings are more alike inside a super-synset than across super-synsets.

C Implementation Detail

C.1 A Contrastive Learning View of Arbitrary Class Pairs

Another way of looking at SimLAP is from the perspective of contrastive learning. SimLAP can be seen as extending contrastive learning from similar sample pairs to arbitrary pairs. Algorithm 1 outlines our framework for arbitrary class-pair contrastive learning. The algorithm begins by sampling

Algorithm 1 SimLAP algorithm.

input: batch size N, constant τ, structure of f_V and f_P for encoding, structure of f_l and f_g for the feature filter.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^N, \{y_k\}_{k=1}^N$ **do**
  $\bar{y} = \text{shuffle}(\{y_k\}_{k=1}^N)$ # Compose random class pairs.
  for all $k \in \{1, \ldots, N\}$ **do**
    $\boldsymbol{h}_k = f_V(\boldsymbol{x}_k)$ # representation
    $\boldsymbol{z}_k = f_P(\boldsymbol{h}_k)$ # features in global space
  end for
  for all $i \in \{1, \ldots, N\}$ **do**
    $\boldsymbol{g}_i = f_g((f_l(y_i) + f_l(\bar{y}_i))/2)$ # shared pair gate of $(y_i, \bar{y}_i)$
    $\bar{\boldsymbol{z}}_{i,k} = \boldsymbol{z}_k \odot \boldsymbol{g}_i \quad \forall k$ # same gate on the anchor and all candidates
    $\bar{s}_{i,k} = \bar{\boldsymbol{z}}_{i,i}^\top \bar{\boldsymbol{z}}_{i,k} / (\|\bar{\boldsymbol{z}}_{i,i}\| \|\bar{\boldsymbol{z}}_{i,k}\|) \quad \forall k$
  end for
  # Positives are other-view samples of the partner class $\bar{y}_i$.
  # The anchor class is excluded unless it is the partner. Other classes are negative.
  define $\ell(i)$ **as**
  $\ell(i) = -\frac{1}{|\mathbb{P}|} \sum_{p \in \mathbb{P}} \log \frac{\exp(\bar{s}_{i,p}/\tau)}{\exp(\bar{s}_{i,p}/\tau) + \sum_{k \in \mathbb{N}} \exp(\bar{s}_{i,k}/\tau)}$
    where $\mathbb{P}$ is the positive set, $\mathbb{N}$ is the negative set.
  $\mathcal{L} = \frac{1}{N} \sum_{k=1}^N \ell(k)$
  update networks f_V, f_P, f_l, and f_g to minimize $\mathcal{L}$
end for
return encoder network $f_V(\cdot)$, and throw away others

mini-batches of input data and their corresponding labels. Each image contributes two augmented views, as stated in Section 3.2. For each anchor i with induced class pair $(y_i, \bar{y}_i)$, the filter produces a single pair-conditioned gate $\boldsymbol{g}_i$. The encoder f_V and projector f_P map every sample to a global feature; the *same* gate $\boldsymbol{g}_i$ is then applied to the anchor and to all candidates before cosine similarities are computed. The contrastive loss is calculated using these similarities and a temperature parameter τ. This loss imposes similarity to the partner class on the coordinates weighted by the gate, and pushes samples from other classes away. The gate weights coordinates for that pair. It does not identify those coordinates as shared visual attributes. The networks f_V, f_P, f_l, and f_g are updated to minimize this loss. After training, only the encoder network $f_V(\cdot)$ is retained for downstream tasks, discarding the gating mechanism used during training.

Pair construction. An arbitrary pair is a class pair, obtained from two randomly selected samples. We implement this by shuffling the samples (equivalently, their labels) within each mini-batch and pairing each sample with the sample at the shuffled index, as in Algorithm 1. The procedure follows the empirical class distribution of the batch rather than sampling uniformly from the $C(C+1)/2$ class-pair subspaces; some subspaces may never appear. Shuffling also guarantees that every sample is paired exactly once per batch.

C.2 Class Similarity

There are two types of class similarity calculation: micro-class similarity and macro-class similarity. Micro-class similarity is a measure that focuses on the similarity between individual instances within two classes, capturing the relationships between the elements from the class pair. Specifically, the micro-class similarity between two classes y_1 and y_2 is defined as the expected value of the cosine similarity $\langle \cdot, \cdot \rangle$ between pairs of instances from these classes:

$$\text{Micro-Sim}(y_1, y_2) = \mathbb{E}[\langle z_i, z_j \rangle], \quad z_i \in y_1,\ z_j \in y_2. \tag{8}$$

This formulation ensures that the similarity is computed directly from the pairwise interactions of instances, providing a detailed and granular measure of how closely the elements of the two classes are related.

In contrast, macro-class similarity provides a more aggregate measure by comparing the expected values (or means) of the instances within each class. This approach captures the overall similarity

Table 12: **Pretraining details**. The output size of Vision Encoder depends on the backbone (2048 for ResNet-50). Learning rate is scheduled by a cosine function with warmup. We use two configurations: without momentum encoder for SupCon, SimCLR, and SimLAP; with momentum encoder for joint loss.

(a) **Hyperparameter**.

With momentum	✗	✓
Batch size	4096	1024
Epochs	200	300
Warmup	12	40
Optimizer	LARS	AdamW
LR	6.4	0.3
Weight decay	1e-6	1.5e-6
Temperature	0.1	0.1

(b) **Projector Structure**.

Layer	Output
BatchNorm1d, ReLU	2048
Linear	2048
BatchNorm1d, ReLU	2048
Linear	256
LayerNorm	256

(c) **Filter Structure**.

Layer	Output
Label Embedding	512
BatchNorm1d, ReLU	512
Linear	1024
BatchNorm1d, ReLU	1024
Linear	256
Sigmoid	256

Table 13: Pretraining cost for 200 epochs on 16 GPUs (2 nodes $\times$ 8 MI250X). GPU-hours $=$ $200 \times 312 \times (\mathrm{s/step}) \times 16/3600$.

Method	GPUs	s/step	GPU-hours
SimCLR	16	1.18	327
SupCon	16	1.15	319
SimLAP	16	1.18	327

between the central tendencies of the classes, rather than focusing on individual instances. The macro-class similarity between y_1 and y_2 is defined as:

$$\text{Macro-Sim}(y_1, y_2) = \langle E[z_i], E[z_j] \rangle, \quad z_i \in y_1,\ z_j \in y_2 \tag{9}$$

Here, $E[z_i]$ and $E[z_j]$ represent the expected values (or means) of the instances in classes y_1 and y_2, respectively. By comparing these expected values, macro-class similarity offers a higher-level view of the relationship between the classes, which can be particularly useful when the overall trends or central tendencies are of primary interest.

C.3 Training Detail

Experiments were conducted on a supercomputer cluster. Each node has one 64-core AMD EPYC 7A53 (Trento) CPU, 512 GiB of DDR4 memory, and four AMD Instinct MI250X accelerators (CDNA 2). Table 13 reports the cost of the pretraining process.

IN1K Pretraining. We conducted extensive experiments on the ImageNet-1K (IN1K) dataset. Our experimental settings are summarized in Table 12a. The data augmentation techniques utilized in this work include random horizontal flipping, color jittering (brightness=0.4,contrast=0.4, saturation=0.4, hue=0.1), normalization to ImageNet statistics, random grayscaling (p=0.2), and Gaussian blurring. To accelerate the training procedure, we apply ESSL (Wu et al., 2024) and amp with half-precision.

IN1P Pretraining. The IN1K pretraining comprises 0.5M pairs, and our implementation cannot exploit every pair. To evaluate a well-trained SimLAP, we consider a small dataset, IN1P (Section G), to study how to effectively use class similarity. The model was trained on IN1P for 20,000 epochs with a batch size of 512. The backbone is ResNet-50.

IN25 Pretraining. We conduct experiments on IN25 to study the impact of distinct pairs in representation learning. To control the semantic distance of sample pairs, we first calculate the macro-class similarity using CLIP (ViT-B/16). Then, we construct positive sets according to the rank of class similarity. *Note:* CLIP (ViT-B/16) is used *solely as an external oracle* to rank the semantic distance between class pairs for the purpose of this analysis; it is never part of the SimLAP training pipeline or learning signal. All models are trained for 1000 epochs.

Transfer Learning via Linear Evaluation. The probe is an MAE linear classifier (He et al., 2022) on a frozen ResNet-50 backbone after normalization. The LARS optimizer (You et al., 2017) was

Table 14: Learning stability of SimLAP on ImageNet with 3 seeds for 100 epochs. Use kNN evaluation with k=10.

	CF10	STL10	Pets	Flowers	DTD	Food	Cars	Aircraft	AVG
Seed 0	82.50	97.11	88.20	62.86	56.54	44.99	26.65	25.77	60.58
Seed 1	82.76	97.16	89.64	63.00	57.34	45.20	25.59	26.67	60.92
Seed 2	82.65	97.19	89.59	63.91	59.41	45.33	27.51	26.22	61.48
Mean	82.64	97.15	89.14	63.26	57.77	45.17	26.58	26.22	60.99
STD	0.13	0.04	0.82	0.57	1.48	0.17	0.96	0.45	0.45

utilized to minimize the softmax cross-entropy loss. No data augmentation techniques were applied, aside from random cropping and resizing images to 224x224 pixels. No regularization methods, such as weight decay, were employed. The learning rate is 1.6.

Transfer Learning via Finetuning. A linear classifier was attached to the vision backbone. The AdamW optimization algorithm was utilized to minimize the objective. The learning rate was dynamically changed by the cosine schedule. The maximum learning rate was determined by the base learning rate (following 1e-4 × BatchSize/256). Several regularization techniques were applied to prevent overfitting, including strong data augmentation, weight decay, and Mixup (Zhang et al., 2018). The data augmentation consists of RandCropResize(224, scale=(0.08,1)), RandomHorizontalFlip(p=0.5), AutoAugment(rand-m9-mstd0.5-inc1) (Cubuk et al., 2018), and RandErase(p=0.25) (Zhong et al., 2020). The images underwent mixup or cutmix (mixup_alpha=0.8, cutmix_alpha=1, label_smoothing=0.1). The weight decay was set to 2e-5. These settings may not be optimal, but we use the same settings for all datasets except the training epochs. We set training epochs to 90 for IN1K, 200 for Food, 1000 for the rest.

Joint Loss. The model was trained using a ViT-Tiny backbone with a batch size of 1024. The AdamW optimizer was employed with a learning rate of 1.5e-4 and weight decay of 0.1. The training ran for 300 epochs, with a linear warmup during the first 40 epochs. Gradients to the patch embedding stem convolution were stopped. MoCo momentum used a cosine schedule, with a contrastive temperature of 0.2.

Mid-training Stage. We use ViT-Base/16 as the backbone in this experiment. Starting from an SSL checkpoint, the pretrained model is further trained with SimLAP or cross-entropy for 100 epochs (batch size 1024, cosine LR schedule). Then, the model is fine-tuned on IN1K with cross-entropy for 100 epochs. Therefore, the pretrained model receives 200 epochs in total with mid-training or 100 epochs without mid-training.

C.4 Baseline Model

We use pretrained models from official repositories: MoCo (He et al., 2020)[1], DINO (Caron et al., 2021)[2], GPaCo (Cui et al., 2023)[3], BarlowTwins (BT) (Zbontar et al., 2021)[4], SimSiam (Chen & He, 2021)[5]. We reproduce the results of SimCLR and SupCon by training them for 200 epochs (see Section C.3).

D Learning Stability

We examine the learning stability of SimLAP by training on ImageNet with different seeds for 100 epochs. Table 14 shows the top-1 accuracy of kNN evaluation with k=10 on 8 datasets. Most of benchmarks have a small standard deviation (less than 1%) except for DTD, indicating the learning is stable.

[1]https://github.com/facebookresearch/moco

[2]https://github.com/facebookresearch/dino

[3]https://github.com/dvlab-research/Parametric-Contrastive-Learning

[4]https://github.com/facebookresearch/barlowtwins

[5]https://github.com/facebookresearch/simsiam

Table 15: Image-retrieval results (mAP) on two sets (Hard and Medium) of ROxford5k and RParis6k (Radenović et al., 2018).

	Medium (M)				**Hard (H)**			
	CE	SupCon	SimCLR	SimLAP	CE	SupCon	SimCLR	SimLAP
ROxford5k	27.13	26.28	22.47	**30.82**	9.22	6.60	4.86	**9.48**
RParis6k	53.41	49.35	46.27	**54.29**	24.80	**28.31**	20.65	27.13

E More Evaluation

Finetuning. Table 16 is a sanity check of the gap to published models after finetuning. SimLAP is finetuned from our pretraining. The other rows use checkpoints from their official repositories under different recipes and epoch budgets. MoCo reaches 91.1 average top-1 and SimLAP reaches 90.2, a gap of 0.9. Note that this is not a matched comparison, as the other rows are not trained with the same recipe as SimLAP.

Table 16: Sanity check via **finetuning** on eight classification tasks. Rows other than SimLAP are published checkpoints, not the recipe in Table 4.

top-1	STL	CIFAR10	Food	Aircraft	Cars	DTD	FLW	Pets	AVG
GPaCo	97.9	**98.3**	89.1	89.3	92.8	70.4	94.1	92.2	90.5
SimSiam	94.9	96.7	88.7	**91.1**	**93.7**	68.5	92.2	90.1	89.5
BT	**98.3**	**98.3**	**89.4**	87.2	92.2	71.9	94.3	92.0	**90.4**
SimCLR	97.7	97.9	89.1	89.8	93.0	72.4	95.3	91.1	90.8
DINO	97.9	97.9	90.3	88.6	93.2	**73.1**	**95.7**	91.7	91.0
MoCo	97.7	98.1	89.7	90.2	93.6	72.0	94.6	**93.3**	91.1
SimLAP	96.6	97.8	88.7	89.9	93.6	68.9	95.4	90.5	90.2

Image Retrieval. Retrieval is an important application of representation learning to find images for a query image (Radenović et al., 2018). As shown in Table 15, SimLAP outperforms other baselines except for the hard set of RParis6k. These results confirm that the cross-class features encouraged by SimLAP produce more general global descriptors, leading to superior image-search quality.

F Hyperparameter

Table 17: **Tuning hyperparameters**. $\underline{*}$ denotes the default setting for tuning hyperparameters. We report Top-1 accuracy of the linear probe (IN1K) on IN1K and kNN ($k = 10$) on CIFAR10. Layer Normalization is vital to train SimLAP. LARS and a two-layer MLP projector are used in the reported runs. The temperature of those runs is $\tau = 0.1$; the underlined cell is that setting, not the best cell of this 100-epoch sweep.

(a) **Normalization**.

Norm	IN1K	CF10
<u>LN</u>	**70.6**	**84.1**
BN	2.2	18.5
None	NaN	NaN

(b) **Optimizer**.

opt	IN1K	CF10	act.
AdamW	70.5	82.9	192
<u>LARS</u>	**70.6**	**84.1**	83

(c) **Dimensions** of features.

Dim	IN1K	CF10	act.
<u>256</u>	70.5	82.9	192
2048	70.3	82.9	306
4096	**70.6**	**84.1**	437

(d) **Temperature** of CL.

τ	IN1K	CF10	act.
0.05	**70.6**	**84.11**	70
<u>0.1</u>	69.8	82.9	108
0.15	70.5	82.9	192

We study the impact of hyperparameters for training. Each experiment was trained on IN1K for 100 epochs. The batch size was 1024. We apply the linear evaluation (Chen et al., 2020) on IN1K and kNN (K=10) on CIFAR10 to assess the learned representations. Table 17 demonstrates the effects of various hyperparameters. 'act.' calculates the average sum of gate vectors for class pairs, indicating the size of subspace for each class pair. Default settings for training are highlighted in underlined.

Normalization. We find that normalization at the end of the projector is critical to train SimLAP. Without normalization, the training process becomes unstable, resulting in NaN errors. Conversely, the model barely converges with Batch Normalization (Ioffe & Szegedy, 2015). This is because Batch Normalization enforces each dimension to be informative (high variance), conflicting with our filter module, which selectively eliminates certain dimensions. Our model only functions effectively with Layer Normalization (Ba et al., 2016).

Training Parameter. Similar to other CL methods (Chen et al., 2020; He et al., 2020), SimLAP's performance is influenced by many factors.

Table 17b compares optimizers. AdamW (Loshchilov & Hutter, 2019) is more efficient in training the neural networks. However, LARS (You et al., 2017) converges to a better solution by about 1.2 percentage points on CIFAR-10. Optimization affects the average activation.

Table 17c illustrates the effect of varying the projector's dimensions. While a large projection space is crucial for transfer learning, increasing dimensionality has minimal impact on linear probing. This is because 256 dimensions suffice for major visual representations, but minor visual information requires more dimensions.

We examined three temperature values (see Table 17d). The reported runs use $\tau = 0.1$. In this 100-epoch sweep, $\tau = 0.05$ scores higher, so the underlined cell is the setting of the main recipe rather than the best cell here. Average activation increases with τ.

G BENCHMARK

IN1P. IN1P is a small ImageNet subset for prolonged training: ten dog breeds (Chihuahua, toy terrier, Walker hound, English foxhound, Saluki, Chesapeake Bay retriever, Rottweiler, Doberman, boxer, Great Dane). Figure 10 shows one image from each class. The subset is a long-training test, not evidence that the gate identifies shared visual attributes.



Figure 10: Random samples from IN1P.

IN25. We introduce IN25, a carefully curated subset of ImageNet-1K designed to study how the semantic distance of positive pairs affects contrastive learning. The dataset comprises 25 classes organized into 5 super-classes: cars, snakes, birds, dogs, and cats, with each super-class containing 5 sub-classes (Figure 11). This hierarchical structure creates multiple levels of semantic relationships: Intra-super-class: Classes within the same super-class (e.g., different dog breeds) exhibit high semantic similarity; Inter-super-class: Classes across super-classes demonstrate varying degrees of semantic distance (e.g., dogs are semantically closer to cats than to cars). To quantify these semantic relationships, we analyze class similarities using CLIP (ViT-B/16) embeddings. Figure 12a presents

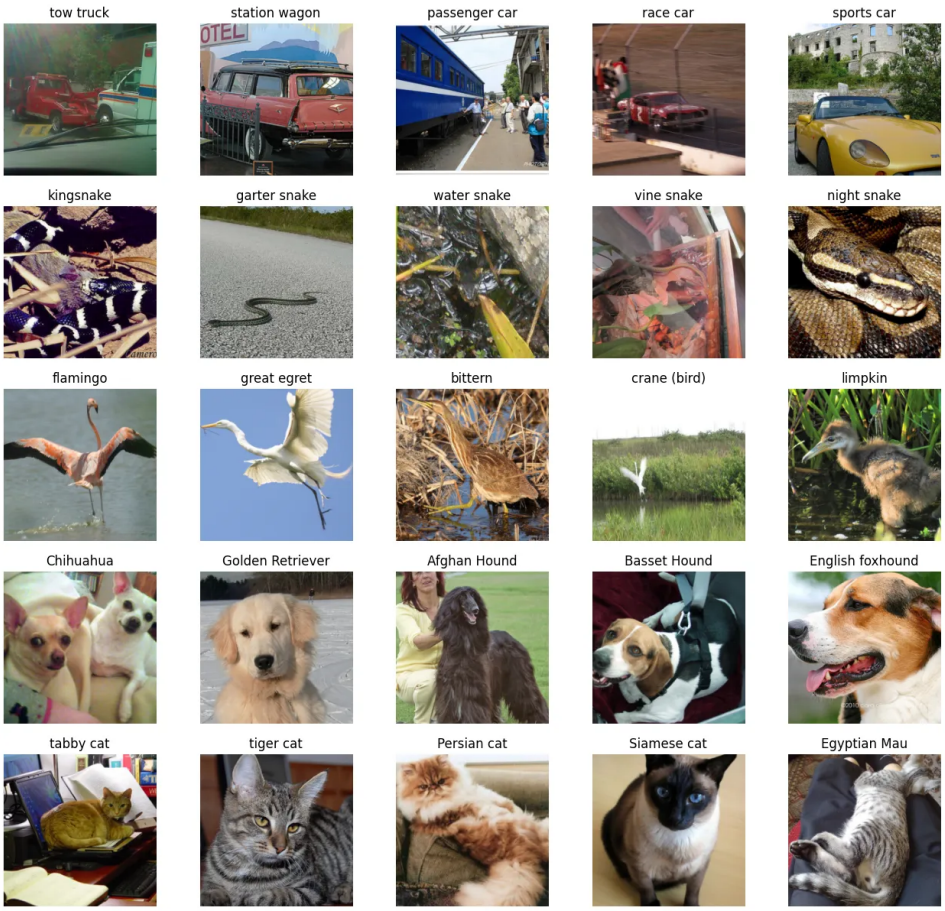
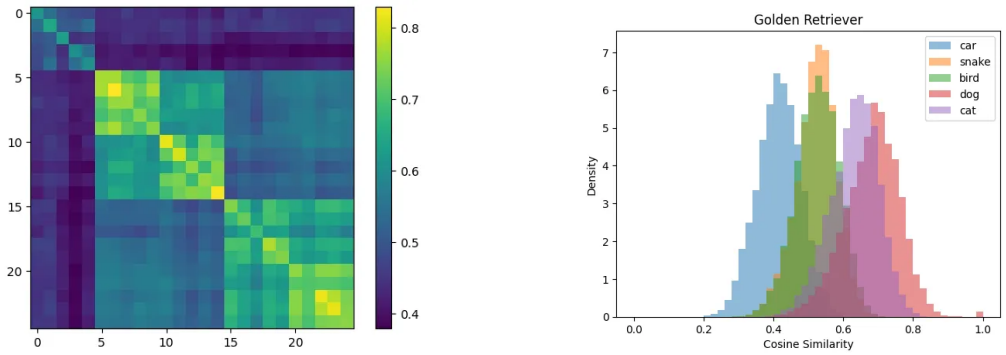


Figure 11: Random samples from IN25.



(a) Class similarity

(b) Similarity to Golden Retriever.

Figure 12: IN25 demonstrates hierarchical semantic distances between classes.

a heatmap of average similarities between all class pairs, revealing clear block-diagonal patterns that correspond to super-class groupings. For a more detailed view, Figure 12b shows similarity distributions relative to a single class (Golden Retriever), demonstrating how semantic distances vary continuously across different super-classes.

Downstream Tasks. We use 8 downstream tasks to evaluate the transfer learning performance: **CIFAR-10**: A widely-used dataset for image classification, consisting of 60,000 32x32 color im-

ages across 10 classes, with 6,000 images per class. It includes common objects such as airplanes, automobiles, birds, and cats. The dataset is split into 50,000 training images and 10,000 test images. **STL-10**: Inspired by CIFAR-10 but designed with unsupervised feature learning in mind. It contains 5,000 labeled images across 10 classes and 100,000 unlabeled images. The images are larger (96x96) and of higher quality compared to CIFAR, making it more challenging and realistic. **Stanford Cars**: A fine-grained visual classification dataset containing 16,185 images of 196 classes of cars. The data is split into 8,144 training images and 8,041 testing images. **Oxford-IIIT Pet Dataset**: Consists of 7,349 images of cats and dogs across 37 breeds. The dataset features 12 cat breeds and 25 dog breeds, with roughly 200 images per class. It's commonly used for tasks such as fine-grained classification and segmentation. The data is split into 3,680 training images and 3,669 testing images. **FGVC-Aircraft**: A benchmark dataset for the fine-grained visual categorization of aircraft. It contains 10,200 images of aircraft, with 100 images for each of 102 different aircraft model variants, most of which are airplanes. The models are organized in a four-level hierarchy: Model, Variant, Family, and Manufacturer. The data is divided into three equally-sized training, validation, and test subsets. **Describable Textures Dataset (DTD)**: An evolving collection of textural images in the wild, annotated with a series of human-centric attributes, inspired by the perceptual properties of textures. The DTD consists of 5,640 images, organized according to a list of 47 terms (categories) inspired by human perception. There are 120 images for each category, with image sizes ranging between 300x300 and 640x640. The images contain at least 90% of the surface representing the category attribute. The data is split into three equal parts for training, validation, and testing, with 40 images per class for each split. **Food-101**: A dataset for food image classification, consisting of 101 categories, with 101,000 images. Each class includes 250 manually reviewed test images and 750 training images, intentionally left uncleaned to include some noise, which manifests as intense colors and occasional incorrect labels. To standardize the images for processing, all images have been rescaled to have a maximum side length of 512 pixels. **Oxford Flowers-102**: A fine-grained image classification dataset comprising 102 flower categories. It contains 8,189 images in total, with each class consisting of between 40 and 258 images. The training set has only 2,040 samples. The dataset is particularly challenging due to the fine-grained nature of the categories and the large variations in scale, pose, and lighting conditions. These tasks involve general image classification tasks and fine-grained classification tasks, providing a comprehensive evaluation of the learned representations for the image.

ROxford5k and RParis6k. In the revisited ROxford5k and RParis6k benchmarks (Radenović et al., 2018), each query-associated positive image is annotated with one of three difficulty labels——Easy, Hard, or Unclear——according to the magnitude of viewpoint, illumination, and occlusion variation relative to the query crop. The Medium evaluation protocol treats all images marked Easy or Hard as positives, whereas the Hard protocol regards only the Hard subset as relevant, ignoring Easy and Unclear instances entirely. Consequently, Medium demands retrieval of both canonical and moderately challenging views, while Hard exclusively rewards localisation under severe appearance changes.