

CAN MODELS LEARN FROM ARBITRARY PAIRS?

Anonymous authors

Paper under double-blind review

ABSTRACT

Representation learning traditionally follows a simple principle: pull semantically similar samples together and push dissimilar ones apart. This principle underlies most existing approaches, including supervised classification, self-supervised learning, and contrastive methods, and it has been central to their success. Yet it overlooks an important source of information: Even when classes appear unrelated, their samples often share latent visual attributes such as shapes, textures, or structural patterns. For example, cats, dogs and cattle have fur and four limbs etc. These overlooked commonalities raise a fundamental question: *can models learn from arbitrary pairs without explicit guidance?*

We show that the answer is yes. The primary challenge lies in learning from dissimilar samples while preserving the notion of semantic distance. We resolve this by proving that for any pair of classes, there exists a subspace where their shared features are discriminative to other classes. To uncover these subspaces we propose **SimLAP**, a **S**imple framework to **L**earn from **A**rbitrary **P**air. SimLAP uses a lightweight feature filter to adaptively activate shared attributes for any given pair. Through extensive experiments we show that models trained via SimLAP can indeed learn effectively from arbitrary pairs. Remarkably, models learned from arbitrary pairs are more transferable than those learned from traditional representation learning methods and exhibit greater resistance to representation collapse. Our findings suggest that arbitrary pairs, often dismissed as irrelevant, are in fact a rich, complementary and untapped source of supervision. By learning from them we move beyond rigid notions of similarity. Hopefully, SimLAP will open an additional pathway toward more general and robust representation learning.

1 INTRODUCTION

The basic tenet of most of the tasks in machine learning, such as classification, clustering, image segmentation, or self-supervised learning, is the notion of similarity or dissimilarity. They both serve to highlight unique characteristics of objects or concepts. While similarities emphasise commonalities, dissimilarities focus on distinctive characteristics. Both notions are essential for understanding and categorising the world around us.

Conventional machine learning models seek representation which aims to maximise the similarity of instances belonging to the same class, cluster or segment, while maximising dissimilarities of samples drawn from different categories. The success is typically measured using cross-entropy (LeCun et al., 1989; Khalid et al., 2023; Sohn, 2016), when the learning is assisted by labels (Khosla et al., 2020) or some other contrastive learning loss function, when labels are not available (Chen et al., 2020; He et al., 2020; Atito et al., 2021). During training, the models fit weight vectors to the respective groups of data so that samples from the same group are pulled together, while samples from different classes are pushed apart (Liu et al., 2016; Wu & Wang, 2020). This objective focuses on discrimination of one group against all the others, and implicitly promotes orthogonal representations (Ranasinghe et al., 2021). However, this one against all approach ignores the potential relationships between different groups, even though it is evident that classes may share common features. For example, cats and dogs are distinct classes but share mammalian traits, such as four limbs, fur, and tail. Even for distinct classes, snakes and ropes both follow elongated curves. The conventional one-against-all learning fails to consider these relationships.

Recently, a resurgence of work has been aware of the importance of common features across classes (Cabannes et al., 2023b; Sobal et al., 2024; Ridnik et al., 2021). These interesting methods apply

additional sources of information to build a better definition of class similarity. However, it is infeasible to provide this Aristotelian annotation for all class pairs because of their tremendous number in datasets, such as ImageNet-1K (Deng et al., 2009). Secondly, such auxiliary information, in the form of e.g. semantic hierarchy (Ridnik et al., 2021), has limitations in annotating the relationship between classes — birds and planes are similar in body shape but semantically distant from each other.

In view of these comments, we propose a completely new paradigm of learning referred to as SimLAP (a Simple framework for Learning from Arbitrary Pairs). Instead of maximising within group and minimising between group similarity, we aim to optimise the *between group similarity which is distinct for each pair of groups (classes)*. This will automatically identify features that are shared by pairs of classes. In the context of our proposition, the first fundamental question is how such learning can be formulated. If we take any pair of groups or classes, we still want them to be distinguishable, that is to be dissimilar, at the end of learning. Hence, any similarity between the pair can possibly exist only in a subspace. Thus our desire to learn similarity between any pair of distinct classes can be realised only by identifying the appropriate subspace for each pair. In conclusion, our learning problem has to be stated as one of searching for subspaces where the similarity of the corresponding class pairs is maximised.

The proposed formulation of learning is fundamentally different from the conventional one-against-all approach pursued to date. It is directed to provide much richer information about the unique groups of data in the training set. However, it raises a second fundamental question: Can it learn to separate each individual group from all the other groups as well? In other words, if we learn that a pair of groups, e.g. dogs and cats have shared features (four limbs), will we be able to separate the groups as well. We do not have a theoretical underpinning for this, but we will demonstrate experimentally in Sections 4 and 5 that this is a viable learning method, that achieves both objectives. It leads to one-against-all separability, as well as providing much richer information about the relationship of any pair of classes in our training set. This contrast with the supervised learning methods that do attempt to reflect class relationships, e.g., Supcon (Khosla et al., 2020), which fail to learn orthogonal representations when assembling distinct pairs as shown in Section 5 Figure 6.

We demonstrate our methodology of learning from arbitrary pair on IN1K. Its 1,000 classes comprise 500,500 (0.5M) pairs, with no explicit annotation regarding their pairwise similarity. To find the requisite class-pairwise subspaces, we introduce a simple module, *feature filter*, to create a subspace for each pair. For a chosen class pair, this adaptive filter learns a gate vector (values between 0 and 1) to identify the subspace in which their samples are discriminative to other class pairs. Our model SimLAP can learn visual representations and subspaces for class pairs simultaneously by gradient descent. We show that SimLAP achieved competitive performance with the models trained conventionally, using the one-against-all approach based on separating similar samples from the rest. We will also discuss how the proposed approach can be combined with other forms of supervision, such as text (Radford et al., 2021) and attributes (Ma et al., 2021).

Before discussing the methodology for the proposed learning method, in Section 3.1 we provide experimental evidence in support of our motivation for learning the similarity (common features) of pairs of distinct classes. The methodology is introduced in Section 3.3. We train our model on IN1K and demonstrate the following experimental findings: SimLAP successfully promotes compactness of distinct pairs in subspaces, while preserving class separability in global space, cf. Section 4.1. Representations learned from arbitrary pairs are competitive or beat the one-vs-all approach in many tasks, cf. Section 4.2. SimLAP is driven by a learning dynamic of representation learning and subspace optimization, cf. Section 5. Furthermore, SimLAP exhibits amazing properties in preventing collapse cf. Figure 7. Finally, we argue that SimLAP is complementary to existing learning mechanisms and validate this by applying SimLAP in joint loss and mid-training, cf. Section 6. In summary, our contributions are as follows: ❶ We propose a novel paradigm of learning from pairs of samples drawn from an arbitrary pair of classes that groups them in their corresponding subspace. ❷ We show that the models learned from arbitrary pairs identify the features shared by each pair. ❸ Most importantly, the models learn to separate individual classes from all the other classes. ❹ Interestingly, the models exhibit better transfer learning performance than one-vs-all-approaches.

2 RELATED WORK

Supervised Learning. Existing visual representation learning (VRL) paradigms, whether supervised learning (SL) or self-supervised learning (SSL), follow the intuition that learning objectives should promote a predefined similarity structure (Cabannes et al., 2023a). The representations implicitly learn the notion of similarity by pulling semantically identical pairs closer while pushing semantically non-identical or distinct pairs away as a whole either directly or indirectly (Liu et al., 2016; Wang et al., 2016). Despite the prevailing success of these VRL paradigms, they lack a systematic mechanism to explicitly learn from semantically distinct pairs, regardless of the semantic distance between these distinct pairs. Recent work finds that semantically distinct but visually similar samples can help in representation learning (Cabannes et al., 2023a; Sobal et al., 2024), as even dissimilar pairs may share semantic attributes (Tsai et al., 2021). Hierarchical representation learning solves this problem by using a semantic hierarchy to capture the shared attributes between different classes (Ridnik et al., 2021; Mohammed & Umaashankar, 2018). Positive Active Learning (PAL) (Cabannes et al., 2023a) reveals that the learning process of both SL and SSL can be expressed in terms of a similarity graph. \mathbb{X} -CLR (Sobal et al., 2024) utilizes the text caption to facilitate the calculation of the similarity graph. Although these work has begun to explore more complex relationships between samples, the requirements of additional annotation or labelling limit their applications. Labels in SimLAP serve a distinct purpose compared to traditional supervised learning methods like Liu et al. (2017); Morin & Bengio (2005); Murtagh & Contreras (2012). Instead of assigning each sample to a single corresponding cluster (Liu et al., 2017), our approach uses labels to select relevant features.

Common Features across Classes. Some CL methods extract common features across classes by using extra variables, such as auxiliary information (Tsai et al., 2021) and attributes (Ma et al., 2021). Nevertheless, the above methods do not address the challenge of learning from distinct samples, e.g. snake-lamp. For example, hierarchical classification methods (Deng et al., 2014; Yan et al., 2015) utilize a predefined hierarchy to capture the shared attributes between different classes. However, constructing such a hierarchy requires significant human effort and domain knowledge. Essentially, these methods still rely on human knowledge to define the relationships between classes. Cross-class features have been shown to mitigate the collapse of decision boundaries (Kim et al., 2024), yet this exploration remains limited in incremental learning. Although common features across samples bring insights on new pretext tasks, such as cross-view completion (Weinzaepfel et al., 2022) and matryoshka representation (Kusupati et al., 2022), introducing this principle into CL is challenging.

3 ARBITRARY PAIR LEARNING

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 optimize similarity between arbitrary pairs of groups, without assuming any predefined semantic relationships or inter-class structure. A key challenge is how to formulate the learning, particularly in the absence of supervision indicating which group pairs should be similar. To address this, we introduce a subspace-based similarity optimisation framework, where learning is constrained to subspaces defined by selected group pairs. Let us dive into motivation of learning in subspaces.

3.1 MOTIVATION FOR LEARNING IN SUBSPACES

A fundamental property of learning is that any pair of classes should be distinguishable from each other. Therefore, maximisation of similarity between arbitrary pairs should only be done in a subspace. Before formulating our approach we would like to investigate whether any similarity between arbitrary pairs in sub-spaces exists for current learning algorithms. To demonstrate we choose pre-trained ResNet50 using supervised learning on IN1K from timm (Wightman, 2019). We extract the class vectors for all the samples in IN1K validation set corresponding to two distinct classes ‘garter snake’ and ‘table lamp’, which have no apparent common features. As shown in Figure 1 using distribution of cosine similarity, they are nearly orthogonal in the global (full) space, yet close in their discriminative subspace. Our subspace selection, picking features that have the same sign, shows similarity for arbitrary pairs in selected subspaces. The calculation of class similarity distribution

can be found in Appendix B.2. This means that our learning can be achieved by learning/optimising for subspaces where the similarity of the corresponding arbitrary class pairs is maximised.

Observation: For high-dimensional orthogonal representations, we can find a subspace for an arbitrary class pair such that their samples are close to each other in that subspace and far away from samples of other classes.

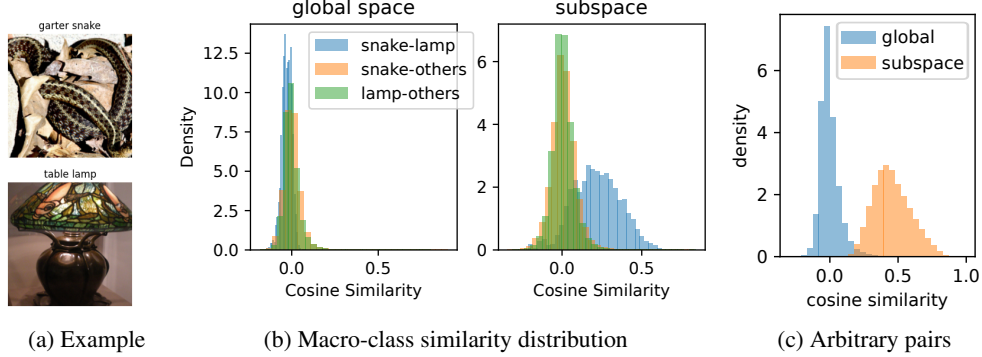


Figure 1: Empirical evidence of discriminative subspaces for ResNet50 trained with SL. Two distinct classes snake-lamp (a) are nearly orthogonal in global space but closer to each other than other classes in a subspace (b). Micro-class similarity distribution of all pairs in global space or subspaces (c). Calculation cf. Appendix B.2

3.2 CONTRASTIVE LEARNING

We revisit the basics of contrastive learning, which promotes the discovery of discriminative features between positive and negative samples. The key to contrastive learning is defining positives and negatives samples. For each anchor sample in the dataset $\mathbf{x} \in \mathcal{X}$, a conventional contrastive learning method defines its positives \mathbf{x}^+ and negatives \mathbf{x}^- according to its learning principles. An encoder network $f_V(\cdot)$ is applied to map the images to a representation vector. A projection network f_P is critical for improving transfer learning performance (Chen et al., 2020) and mitigating dimensional collapse (Jing et al., 2022). Overall, features are extracted by $\mathbf{z} = f_P(f_V(\mathbf{x}))$. The objective helps to identify the features that can separate positives and negatives, which is achieved by InfoNCE (Oord et al., 2018):

$$\mathcal{L} = -\log \frac{\exp(\langle \mathbf{z}, \mathbf{z}^+ \rangle / \tau)}{\exp(\langle \mathbf{z}, \mathbf{z}^+ \rangle / \tau) + \sum_{\mathbf{z}^- \in \mathcal{Z}^-} \exp(\langle \mathbf{z}, \mathbf{z}^- \rangle / \tau)}, \quad (1)$$

where $\langle u, v \rangle = \frac{u \cdot v}{\|u\| \|v\|}$ denotes the cosine similarity between two vectors.

3.3 LEARNING FROM ARBITRARY PAIRS

SimLAP maximises the similarity between two samples drawn from an arbitrary pair of classes, constrained to their corresponding subspace. In other words, the learning objective seeks subspaces where the similarity in terms of common features between selected arbitrary class pairs is maximised. The overall architecture is depicted in Figure 2. We randomly pair two samples \mathbf{x}_i and \mathbf{x}_j in a batch belonging to any class pair y_i - y_j and select their corresponding subspace by an *adaptive feature filter*. The subspace can be sampled by shuffling cf. Appendix B.1. In this subspace, the samples from y_i - y_j comprise positive pairs and the samples from any other classes comprise negative pairs. Specifically, for an anchor sample \mathbf{x}_i from class y_i in a mini-batch, we randomly choose one sample \mathbf{x}_j from the mini-batch belonging to class y_j and treat the sample belonging to y_j as positive, \mathbb{P} . The samples pairs in the mini-batch that do not belong to y_i and y_j pair are negative \mathbb{N} . To maximise the similarity between arbitrary positive pair corresponding to anchor sample \mathbf{x}_i and minimise the similarity between negative pairs we adapt infoNCE loss. The loss function for the

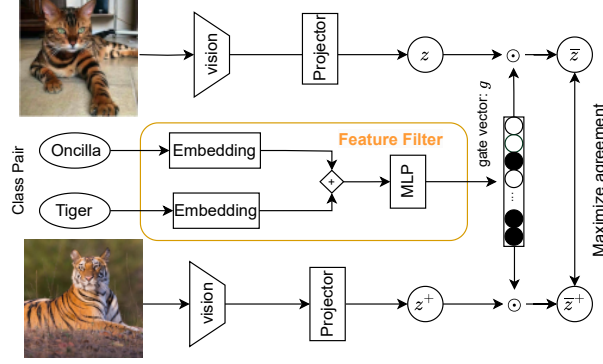


Figure 2: **Learning from Arbitrary Pairs.** SimLAP learns representations by maximizing the agreement of common features between arbitrary classes in a subspace, cf. Appendix B.1.

anchor sample is formulated as follows: $\mathcal{L}(y_i, y_j) =$

$$-\sum_{\bar{z}^+ \in \mathbb{P}} \frac{1}{|\mathbb{P}|} \log \frac{\exp(\langle \bar{z}, \bar{z}^+ \rangle / \tau)}{\exp(\langle \bar{z}, \bar{z}^+ \rangle / \tau) + \sum_{\bar{z}^- \in \mathbb{N}} \exp(\langle \bar{z}, \bar{z}^- \rangle / \tau)} \quad (2)$$

where $\{\bar{z}, \bar{z}^+, \bar{z}^-\} = g(y_i, y_j) \odot \{z, z^+, z^-\}$ denote the features in the subspace of y_i, y_j . The elements of the gate vector $g(y_i, y_j)$, ranging from 0 to 1, control the activation of each feature.

Note that from a mini-batch size of 100 it is possible to build 5050 arbitrary positive pairs. However, for efficiency and for fairness to other learning methods we only consider 100 positive pairs where each sample in the mini-batch becomes anchor only once.

Feature Filter. The feature filter is a vital component to enable learning from arbitrary pairs. It selectively activates certain dimensions, effectively generating a subspace for the selected class pair, to represent their common features. This module enables the discovery and utilization of shared information between distinct samples. The feature filter is a simple lightweight module consisting of two parts: a label embedding and an MLP network. The label embedding converts discrete labels to continuous vectors. We use the mean of the two label vectors to represent the common information between two classes. The MLP layer generates gate vectors to select the corresponding dimensions for the common information. The calculation is

$$g(y_i, y_j) = \sigma(f_g((f_l(y_i) + f_l(y_j))/2)), \quad (3)$$

where $f_l(\cdot)$ denotes an embedding layer that converts a label to a 512-dimensional vector, $f_g(\cdot)$ denotes MLP network consisting of three linear layers with ReLU non-linearity in between them, and $\sigma(\cdot)$ is the Sigmoid activation function. In this framework, each class is equally pulled to or pushed from other classes in different subspaces, meaning that labels do not promote compact embeddings explicitly for any specific class pair.

All model parameters are randomly initialized to evaluate SimLAP capacity to learn from arbitrary visual pairs using only discrete labels. While label embeddings could be initialized from a pre-trained LLM to leverage semantic priors, including potential similarities between arbitrary class pairs, we intentionally avoid this to isolate the learning ability of SimLAP without external knowledge. Surprisingly, SimLAP learns the similarity between classes directly from visual information, cf. Figure 3. Our analysis reveals that the gate vectors will learn to select the dimensions with the same sign cf. Appendix A.2. Further details of SimLAP can be found in Appendix B.1 with a detail algorithm outlined in Algorithm 1.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTING

We trained SimLAP on ImageNet (Deng et al., 2009) where its 0.5M subspaces are optimized efficiently in only 1M iterations. We choose two baseline methods to compare the effectiveness of

Table 1: **Transfer learning performance with KNN (k=10)** across 8 classification tasks.

top-1	Aircraft	CF10	Cars	DTD	FLW	Food	Pets	STL	AVG
SL	28.5	88.0	31.6	66.5	68.3	58.4	91.1	97.3	66.2
Supcon*	33.7	83.8	30.6	55.0	68.3	53.5	88.8	95.4	63.6
SimLAP	40.5	85.3	34.8	62.0	74.9	55.3	89.2	95.4	67.2

arbitrary pairs compared to identical pairs. The backbone is ResNet50 (He et al., 2016). SL denotes the standard supervised setting with cross-entropy and one-hot vectors. We use the pretrained ResNet-50 from timm (Wightman, 2019). We trained Supcon (Khosla et al., 2020) as described in Appendix B.3. These two methods represent the majority in utilizing labels to promote the compactness of identical pairs. Note that our implementation of Supcon* is not as good as reported in the original paper due to a small batch size (1024) and a long-time training (1000 epochs), because the contrastive framework is sensitive to batch size (Chen et al., 2020). We apply the same training settings for both Supcon* and SimLAP to fairly compare the efficiency of learning from identical pairs and arbitrary pairs.

Embedding Analysis. We visualize the learned embeddings from the backbone network (excluding the projector and filter) by analysing representations on the IN1K validation set. As shown in Figure 5, SimLAP exhibits better statistical character in class separation. We also employ *t*-SNE (Van der Maaten & Hinton, 2008) to visualize the embeddings of 10 classes from the IN1K validation set in Figure 3. The visualization reveals two key findings: (1) In the global space, SimLAP maintains clear class separation as good as the methods optimized in the global space, despite being optimized in subspaces; (2) When examining the subspace for specific class pairs (e.g., Garter snake-Chihuahua), our model successfully identifies shared features that bring these seemingly distinct classes closer while preserving the overall structure of the embedding space. (3) The feature filter captures class similarity in gate vectors and label embedding as shown in Figure 4.

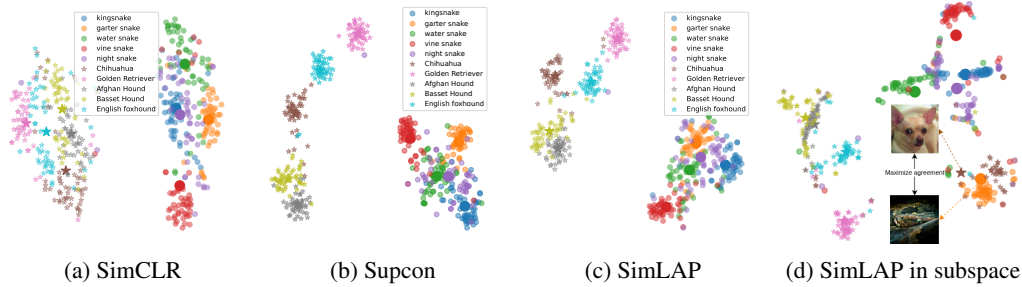


Figure 3: ***t*-SNE visualization** of learned embeddings for 10 classes from two groups: 5 dog breeds (star) and 5 snake species (dot). Large markers indicate class centers, with stars representing dog classes and points representing snake classes. All models separate coarse classes (dogs and snakes). SimCLR struggles to differentiate between dog breeds. (d) highlights the unique ability of SimLAP to bring Chihuahua and garter snake close together within a subspace. Interestingly, optimizing disparate pairs in subspaces results in similar global embedding to supervised one (Supcon).

4.2 TRANSFER LEARNING

Although our framework optimizes representations in subspaces, global representations exhibit good transfer learning performance and class separation. See more evaluations in Appendix C.

Image Classification with K-nearest Neighbour (KNN). We use the KNN protocol to present the transfer learning performance. KNN can largely reveal the learned semantic measurement. As shown in Table 1, SimLAP has a higher average classification accuracy on 8 tasks. Learning from arbitrary pairs extract transferable features.

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 2, SimLAP consistently outperforms two baselines. These results confirm that the intra-class orthogonality encouraged by SimLAP produces

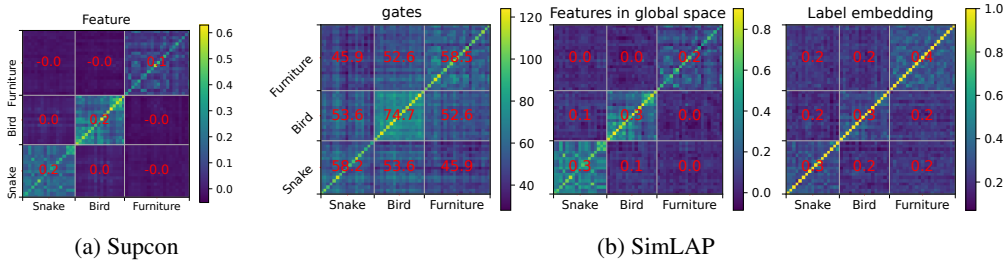


Figure 4: **Class similarity for gates, global features, and label vectors.** Red numbers denote the average similarity within each super synset region (snake, wading bird, furniture). Higher class similarity of super synset indicates that feature filter in SimLAP learns the relationships between classes from representations.

Table 2: Image-retrieval results (mAP) on ROxford5k and RParis6k (Radenović et al., 2018).

	Medium (M)			Hard (H)		
	SL	Supcon*	SimLAP	SL	Supcon*	SimLAP
ROxford5k	27.13	26.28	30.82	9.22	6.6	9.48
RParis6k	53.41	49.35	54.29	28.62	24.8	28.31

more discriminative global descriptors, leading to superior image-search quality without any task-specific re-ranking or query expansion.

Dense Prediction. Classification tasks reflect the quality of global features. To further study the ability to extract subtle information in local representations, we performed semantic segmentation with frozen features. As shown in Table 4, SimLAP has higher mIoU scores, revealing competitive local characteristics.

Domain Shift. Models trained on a dataset may perform poorly when the input distribution changes due to adaptive overfitting. We used several validation sets, different from the original ImageNet dataset, to assess the robustness of the learned representations. For each model, we train a linear classifier using the IN1K training set. We then evaluate the top-1 accuracy on different validation sets to exhibit performance under domain shift. As shown in Table 3, MSimLAP exhibits competitive performance with joint loss of ICL and CCL, which has been proven to solve class collapse (Islam et al., 2021; Xue et al., 2023).

5 MODEL ANALYSIS

Finding 1: Filter makes learning robust to distinct pairs.

The major difficulty of SimLAP is dealing with distinct pairs in which common visual similarity can hardly be found, such as snake-lamp. To study the impact of reducing common features in pairs, we conducted a controllable experiment in increasing the semantic distance of positive pairs. We run the experiment on IN25, a dataset of 25 classes with well-defined semantic relationships (details in Appendix B.3). Each model was trained for 1000 epochs. As shown in Figure 6, the performance of the model without the filter decreases over semantic distance, because distinct pairs define a wrong

Table 3: **Domain shift of IN1K.** We report top-1 accuracy on origin, ImageNet-Sketch (Wang et al., 2019) and ImageNetv2 (Recht et al., 2019) (matched, threshold0.7, and top-images). SL+SSL denotes a combination of SL and MoCo (Islam et al., 2021).

top-1	origin	Scketch	matched	threshold0.7	top-images
SL	76.1	22.4	63.3	72.7	78.1
SL+SSL	76.7	26.1	64.8	73.6	78.3
SimLAP	76.0	29.3	64.1	73.6	78.83

Figure 5: **Inter- and intra- Similarity distribution** on IN1K. The number in the middle denotes the overlap of two distributions. The sharpest inter-class distribution and the minimal overlap indicate that SimLAP learns representations that better preserve class-specific information while maintaining clear class boundaries.

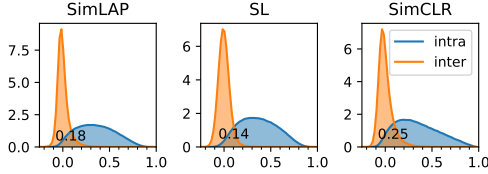
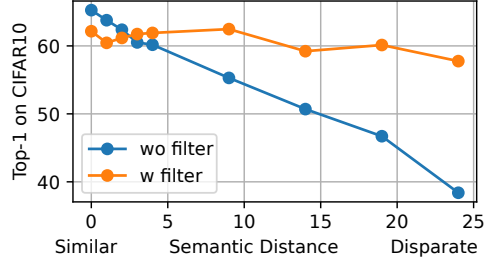


Figure 6: **Adaptive filter is crucial to deal with distinct pairs.** X axis denote semantic distance of the positive set (ranked by CLIP). Models can hardly learn meaningful representations from distinct pairs without the feature filter.



semantic relationship between classes. In contrast, the model with filter is robust in dealing with distinct pairs.

Table 4: **Semantic Segmentation** with frozen features via FCN.

mIoU	cityscapes	voc12aug
SL	52.14	54.9
Supcon*	50.50	43.8
SimLAP	49.92	45.76

Table 5: **Optimizing multiple subspaces in a step.** Runtime (seconds) measures the elapse time to forward 128 samples, tested on a single V100. Report top-1 with KNN on CF10.

Num	GFlops	Runtime	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

Finding 2: SimLAP simultaneously optimises representations from visual data and subspaces for class pairs.

Our model involves two independent learning processes: identifying the subspace for each class pair and learning the common features of the class pair. By default, we randomly select a single subspace to optimize these representations. We investigate the impact of increasing the number of optimized subspaces. Specifically, we first compute global features by forwarding images once, then generate multiple subspaces using Equation (3) to optimize the objective in Equation (2). As shown in Table 5, increasing the number of optimized subspaces enhances performance but increases training time. For simplicity and computational efficiency, we default to optimizing a single subspace for each training step.

Finding 3: The common features across classes are effective in preventing collapse.

To thoroughly investigate the collapse issues, we construct IN1P, a focused subset of IN1K containing 12K images from 10 highly-related dog breeds. This small dataset allows us to train extensively on all possible class pairs. We train models for 20,000 epochs to ensure that models reach minimum. In Figure 7a, we compare the top-1 accuracy with KNN on CIFAR10 across different methods. SimCLR exhibits clear signs of dimensional collapse (Jing et al., 2022) after 3,000 epochs, evidenced by the declining performance and decreasing singular values, from Figure 7b. This reveals that while the non-linear projector can relieve dimensional collapse, it cannot prevent it entirely. Incorporating supervision into CL is a promising solution (Xue et al., 2023). SimLAP demonstrates sustained performance improvement throughout the extended training period, surpassing both Supcon and SimCLR. This robustness against dimensional collapse and class collapse can be attributed to discovering rich common features between classes.

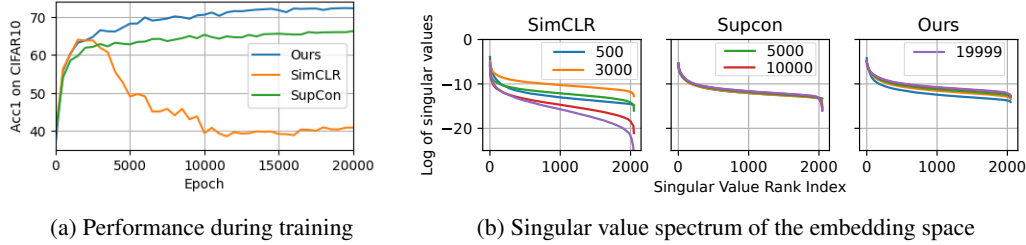


Figure 7: **Preventing collapse by learning common features.** Lower singular values suggest that representations are concentrating information in fewer dimensions. Models trained on IN1P (10 dogs) and evaluated on CIFAR10 show that SimLAP consistently outperforms baseline methods.

6 APPLICATION

Joint Loss. Our learning paradigm is complementary to existing ones, bringing further improvements. To combine SimLAP and MoCov3 (He et al., 2020), we introduce a joint loss by $\frac{1}{1+\beta}\mathcal{L}_{SSL} + \frac{\beta}{1+\beta}\mathcal{L}_{SimLAP}$. From Table 6, the performance is improved by balancing two losses.

Mid-training. Mid-training is a stage between SSL (MAE (He et al., 2022) here) and supervised finetuning (SFT). From Table 7, we can see a significant performance drop for long SFT due to over-adaptation (Hao et al., 2025). SimLAP relieves this problem and gains 0.4% improvement.

Table 6: Joint loss of SimLAP and MoCov3 (ResNet50) improves representations, cf. Appendix B.3.

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 7: SimLAP (ViT-Base) relieves over-adaptation, cf. Appendix B.3. FT and SimLAP train 100 epochs. FT* trains 200 epochs.

training pipeline	epochs	top1
SSL \rightarrow FT	100	83.6
SSL \rightarrow FT*	200	83.1
SSL \rightarrow SimLAP \rightarrow FT	200	84.0

7 DISCUSSION

Our study demonstrates that learning from arbitrary pairs is not only feasible but also beneficial. By introducing SimLAP, we show that even pairs of semantically distant classes can reveal common features when constrained to appropriate subspaces. This challenges the conventional “one-vs-all” paradigm and highlights arbitrary pairs as a complementary source of supervision. The framework consistently preserves class separability while uncovering shared subspaces, leading to stronger transferability and robustness against collapse. To keep the focus on this principle, we adopted a minimal contrastive setup, avoiding additional components, e.g. momentum encoder. While such techniques could further boost performance, our goal was to isolate the effect of arbitrary pairs and make their contribution explicit. The results suggest that arbitrary-pair supervision is especially valuable when identical pairs suffer from diminished signal due to overfitting, or (the second point).

Limitation. Given the large number of possible pairs ($\approx 0.5M$ in IN1K), our 1M iteration training could not fully explore the available space, leaving many informative relations not fully leveraged. Further, not all pairs are meaningful, e.g., “snake-lamp” pair lack obvious commonalities. How to efficiently identify and utilise the valuable pairs without prior annotation remains an open challenge.

Future Work. A promising direction is to extend arbitrary-pair learning with richer forms of supervision. In multimodal data, for example, two images with captions such as “a clock is mounted to a house” and “a clock beside a brick sidewalk on the Baker Street” both highlight the concept of “clock”. Feature filters could exploit this overlap to align disparate visual inputs. Incorporating textual cues, attributes, or hierarchical metadata may further enhance the robustness and transferability of learned representations.

8 ETHICS STATEMENT

We study a generic mechanism for learning representations from arbitrary sample pairs. No private or sensitive information was collected or processed. All experiments rely on publicly available datasets (ImageNet-1K, CIFAR, etc.) or standard splits derived from them. We are not aware of any foreseeable privacy breaches or discriminatory consequences arising from this work.

9 REPRODUCIBILITY STATEMENT

Implementation details, hyper-parameters, and training recipes are documented in Appendix B. Source code and pretrained weights are provided in the supplementary material. All datasets except IN1P and IN25 are publicly downloadable; the latter two can be constructed from ImageNet-1K with the scripts we supply, ensuring full reproducibility. The training code of SimLAP on CIFAR10 is provided.

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 2023a.
- 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*, pp. 16274–16283, 2023b.
- 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.

- Yifan Hao, Xingyuan Pan, Hanning Zhang, Chenlu Ye, Rui Pan, and Tong Zhang. Understanding overadaptation in supervised fine-tuning: The role of ensemble methods. *arXiv preprint arXiv:2506.01901*, 2025.
- 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, Ross Girshick, and Piotr Dollar. Rethinking imagenet pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- 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>.
- Ashraful Islam, Chun-Fu Richard Chen, Rameswar Panda, Leonid Karlinsky, Richard Radke, and Rogerio Feris. A broad study on the transferability of visual representations with contrastive learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8845–8855, 2021.
- 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>.
- Syed Safwan Khalid, Muhammad Awais, Zhen-Hua Feng, Chi-Ho Chan, Ammarah Farooq, Ali Akbari, and Josef Kittler. Npt-loss: Demystifying face recognition losses with nearest proxies triplet. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):15249–15259, 2023. doi: 10.1109/TPAMI.2022.3162705.
- 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.
- Taeheon Kim, Jaeyoo Park, and Bohyung Han. Cross-class feature augmentation for class incremental learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13168–13176, 2024.
- Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.
- Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. doi: 10.1162/neco.1989.1.4.541.
- Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. *arXiv preprint arXiv:1612.02295*, 2016.
- Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 212–220, 2017.

- Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015.
- 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>.
- Martin Q Ma, Yao-Hung Hubert Tsai, Paul Pu Liang, Han Zhao, Kun Zhang, Ruslan Salakhutdinov, and Louis-Philippe Morency. Conditional contrastive learning for improving fairness in self-supervised learning. *arXiv preprint arXiv:2106.02866*, 2021.
- 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.
- Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In *International workshop on artificial intelligence and statistics*, pp. 246–252. PMLR, 2005.
- Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.
- Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- 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.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.
- Kanchana Ranasinghe, Muzammal Naseer, Munawar Hayat, Salman Khan, and Fahad Shahbaz Khan. Orthogonal projection loss. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12333–12343, 2021.
- Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pp. 5389–5400. PMLR, 2019.
- Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.
- 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. *arXiv preprint arXiv:2407.18134*, 2024.
- Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.
- Yao-Hung Hubert Tsai, Tianqin Li, Weixin Liu, Peiyuan Liao, Ruslan Salakhutdinov, and Louis-Philippe Morency. Integrating auxiliary information in self-supervised learning. *arXiv preprint arXiv:2106.02869*, 2021.
- Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
- Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019.

- 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.
- Philippe Weinzaepfel, Vincent Leroy, Thomas Lucas, Romain Brégier, Yohann Cabon, Vaibhav Arora, Leonid Antsfeld, Boris Chidlovskii, Gabriela Csurka, and Jérôme Revaud. Croco: Self-supervised pre-training for 3d vision tasks by cross-view completion. *Advances in Neural Information Processing Systems*, 35:3502–3516, 2022.
- 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.
- Yihao Xue, Siddharth Joshi, Eric Gan, Pin-Yu Chen, and Baharan Mirzasoleiman. Which features are learnt by contrastive learning? on the role of simplicity bias in class collapse and feature suppression. In *International Conference on Machine Learning*, pp. 38938–38970. PMLR, 2023.
- 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. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- 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 THEORY PROOF

A.1 EXISTENCE OF DISCRIMINATIVE SUBSPACES

Our work is based on the hypothesis that there exists a subspace to allow arbitrary pairs to be close while the other classes are far away.

Definition. Let N orthogonal class vectors be sampled from D -dimensional Gaussian distributions: $\mathbf{z} \in \mathbb{R}^{N \times D}$, $\mathbf{z}_i^d \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1/D)$.

Subspace Selection. For an arbitrary pair $(i, j) \in \mathbb{P}^+$, we select the gate vector $\mathbf{g}(i, j) = \mathbf{I}_d = \mathbb{1}_{\mathbf{z}_i^d \mathbf{z}_j^d > 0}$. The chance of each dimension being positive is 0.5: $p(\mathbf{g}(i, j) = 1) = p(\mathbf{z}_i^d > 0)p(\mathbf{z}_j^d > 0) + p(\mathbf{z}_i^d < 0)p(\mathbf{z}_j^d < 0)$. Hence $\mathbf{g}(i, j) \sim \text{Binomial}(D, 0.5)$.

For orthogonal representations, given an arbitrary pair, we can select a subspace with positive sign and have the following:

$$\begin{aligned} \mathbb{E}_{(i,j) \in \mathbb{P}^+} [\bar{\mathbf{z}}_i \bar{\mathbf{z}}_j] &= \frac{1}{\pi}, \\ \mathbb{E}_{(i,k) \in \mathbb{P}^-} [\bar{\mathbf{z}}_i \bar{\mathbf{z}}_k] &= 0, \end{aligned} \tag{4}$$

where $\bar{\mathbf{z}} = \mathbf{z} \odot \mathbf{g}(i, j)$.

Proof. Fix a positive pair $(i, j) \in \mathbb{P}^+$ and write

$$\mathbf{g} = \mathbf{g}(i, j) \in \{0, 1\}^D, \quad \mathbf{g}_d = \mathbb{1}_{\mathbf{z}_i^d \mathbf{z}_j^d > 0}.$$

Because $\mathbf{z}_i^d, \mathbf{z}_j^d \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1/D)$, the product equals to only consider the half part. The coordinate-wise conditional expectation of the product given the gate is

$$\mathbb{E}[\mathbf{z}_i^d \mathbf{z}_j^d \mid \mathbf{g}_d = 1] = \mathbb{E}[|z_i^d| |z_j^d|] = \frac{2}{\pi} \frac{1}{D},$$

because the mean of a standard half-normal is $\mathbb{E}[|X|] = \sqrt{\frac{2}{\pi}}$, $X \sim \mathcal{N}(0, 1)$.

In addition,

$$\Pr(\mathbf{g}_d = 1) = \Pr(\mathbf{z}_j^d > 0) = \frac{1}{2} \quad \text{and} \quad \mathbf{g}_d \stackrel{\text{i.i.d.}}{\sim} \text{Bernoulli}(1/2).$$

Hence

$$\mathbb{E}_{(i,j) \in \mathbb{P}^+} [\mathbf{g}^\top (\mathbf{z}_i \odot \mathbf{z}_j)] = \sum_{d=1}^D \mathbb{E}[\mathbf{g}_d \mathbf{z}_i^d \mathbf{z}_j^d] = \sum_{d=1}^D \Pr(\mathbf{g}_d = 1) \mathbb{E}[\mathbf{z}_i^d \mathbf{z}_j^d \mid \mathbf{g}_d = 1] = D \frac{2}{\pi} \frac{1}{D} \frac{1}{2} = \frac{1}{\pi}.$$

For a negative pair $(i, k) \in \mathbb{P}^-$ the variables \mathbf{z}_i^d and \mathbf{z}_k^d are independent and zero-mean, so

$$\mathbb{E}[\mathbf{z}_i^d \mathbf{z}_k^d] = 0 \quad \text{regardless of the gate.}$$

Stimulation. We simulate the process of feature selection by positive contribution. We generate 10 class vectors from $\mathcal{N}(0, 1/D)$. The gap calculates the difference between macro-class similarity and the largest similarity to negative samples, revealing the representation separability:

$$\text{gap}(i, j) = \mathbf{z}_i \mathbf{z}_j - \max_k \mathbf{z}_i \mathbf{z}_k. \tag{5}$$

Our simulated results match our theoretical analysis as shown in Figure 9.

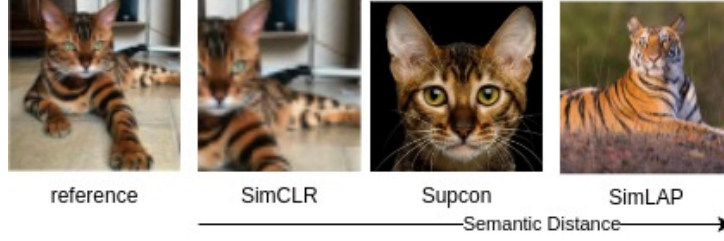


Figure 8: Comparison of sample pairs in three CL methods.

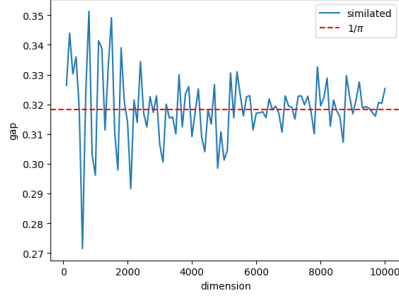


Figure 9: The expectation of the gap between intra-class and inter-class fits the theoretical value.

subspace	label	y2	y1	y3	y3	y2
y2-y2	y2	p	n	n	n	p
y1-y2	y1	p	p	n	n	p
y3-y1	y3	n	p	p	p	n
y3-y3	y3	n	n	p	p	n
y2-y3	y2	p	n	p	p	p

Figure 10: Positive and negative samples in SimLAP. ‘p’ denotes positive, ‘n’ denotes negative.

A.2 GRADIENT ANALYSIS

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 $s(i, j) = \langle \mathbf{g} \odot \mathbf{z}_i, \mathbf{g} \odot \mathbf{z}_j \rangle / \tau$, parameterized by a gate vector $\mathbf{g} = g(y_i, \bar{y}_j) \in \mathbb{R}^D$, \bar{y} is the shuffled labels. For a subspace of class pair $y_i - y_j$ and temperature τ . The objective function L minimizes the InfoNCE loss for two positive pairs:

$$L = -\log \left(\frac{\exp(s(i, j))}{\sum_{k \neq j} \exp(s(i, k))} \right) - \log \left(\frac{\exp(s(j, i))}{\sum_{k \neq i} \exp(s(j, k))} \right). \quad (6)$$

To optimize L , the gradient of L with respect to each dimension d of the gate vector \mathbf{g} is derived as:

$$\frac{\partial L}{\partial g^d} = \frac{2g^d}{\tau} \left[\sum_{k \neq i, j} p(x_k | i) z_i^d z_j^d - (1 - p(x_j | i)) z_i^d z_j^d \right. \\ \left. \sum_{k \neq i, j} p(x_k | j) z_j^d z_i^d - (1 - p(x_i | j)) z_i^d z_j^d \right]. \quad (7)$$

Here, $p(x_k | i) = \frac{\exp(s(i, k))}{\sum_{j \neq i} \exp(s(i, j))}$ denotes the softmax probability of sample k being the class i . The sign of the gradient is governed by the product $z_i^d z_j^d$, with its magnitude adjusted by $\mathbf{g} \in [0, 1]^D$. Specifically, a positive product between positive pairs ($z_i^d z_j^d > 0$) or a negative product between negative pairs ($z_i^d z_j^d < 0$) promotes an increase in g^d . The gate vectors will be nearly binary to find the subspace for an arbitrary pair by activating or inactivating dimensions cf. Figure 15.

B IMPLEMENTATION DETAIL

B.1 CONTRASTIVE LEARNING WITH ARBITRARY PAIRS

Another way of looking at SimLAP is from the prospective contrastive learning. SimLAP can be seen as extending contrastive learning from similar sample pairs to arbitrary pairs. Figure 8

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 $\{\mathbf{x}_k\}_{k=1}^N, \{y_k\}_{k=1}^N$ **do**

$\bar{y} = \text{shuffle}(\{y_k\}_{k=1}^{2N})$ # Compose random class pairs.

for all $k \in \{1, \dots, N\}$ **do**

$\mathbf{h}_k = f_V(\mathbf{x}_k)$ # representation

$\mathbf{z}_k = f_P(\mathbf{h}_k)$ # features in global space

end for

for all $i \in \{1, \dots, N\}$ and $j \in \{1, \dots, N\}$ **do**

$\mathbf{g}_i = f_g((f_l(y_i) + f_l(\bar{y}_i))/2)$ # generate gate vectors.

$\tilde{\mathbf{z}}_i = \mathbf{z}_i \odot \mathbf{g}_i$ # features in subspace

$\bar{s}_{i,j} = \tilde{\mathbf{z}}_i^\top \tilde{\mathbf{z}}_j / (\|\tilde{\mathbf{z}}_i\| \|\tilde{\mathbf{z}}_j\|)$ # pairwise similarity

end for

 # Samples are from y_k or \bar{y}_k are positive.

 # Samples are not from y_i or \bar{y}_i are negative.

define $\ell(i, j)$ **as**

$\ell(i) = -\frac{1}{|\mathbb{P}|} \sum_{y_p \in \mathbb{P}} \log \frac{\exp(s_{i,p}/\tau)}{\exp(s_{i,p}/\tau) + \sum_{k \in \mathbb{N}} \exp(s_{i,k}/\tau)}$

 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

compares the positive pairs in SimCLR, Supcon, and SimLAP. Our algorithm selects positive and negative pairs according to the generated subspace as shown in Figure 10. Algorithm 1 outlines our framework for learning from arbitrary pairs. The algorithm begins by sampling mini-batches of input data and their corresponding labels. Notably, we sample positive pairs from the mini-batch instead of augmented views. For each pair of classes y_{2k-1} - y_{2k} , it generates a subspace representation using the gate function f_g , which takes the average of the label embeddings as input. The encoder f_V extracts representations for both the reference sample and its arbitrary pair. These representations are then projected into the global space using the projector f_P and element-wise multiplication with the gate vector \mathbf{g}_k to project into subspaces. Pairwise similarities are computed for all samples in the batch using cosine similarity. The contrastive loss is calculated using these similarities and a temperature parameter τ . This loss encourages the model to maximize similarity between arbitrary pairs in their shared subspace while minimizing similarity with samples from other classes. 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. This approach allows the model to learn transferable representations that capture common features across seemingly disparate classes.

Subspace Sampling. Each sample can be optimized within multiple related subspaces. Ideally, we uniformly sample one subspace from all available options. However, when the batch size is small, there is a risk of sampling a subspace without positive pairs in the batch. To mitigate this, we can resample until a non-empty subspace is obtained. Alternatively, we employ a simpler strategy by shuffling the labels within a batch. This ensures that at least one positive pair is present in the sampled subspace.

B.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 distance $\langle \cdot, \cdot \rangle$ between pairs of instances from these classes:

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

Table 8: **Pretraining details.** The output size of Vision Encoder depends on the backbone (2048 for ResNet50). The optimizer is LARS. Learning rate is scheduled by a cosine function with warmup.

(a) Hyperparameter.			(b) Projector Structure.		(c) Filter Structure.	
	SimLAP	SimLAP†	Layer	Output	Layer	Output
Batch size	1024	1024	Vision Encoder	-	Label Embedding	512
Epochs	1000	300	BatchNorm1d, ReLU	2048	BatchNorm1d, ReLU	512
Warmup	40	40	Linear	2048	Linear	1024
LR	0.2	0.3	BatchNorm1d, ReLU	2048	BatchNorm1d, ReLU	1024
Weight decay	1e-4	1.5e-6	Linear	256	Linear	256
Runtime (h)	56.3	25.3	LayerNorm	256	Sigmoid	256

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 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:

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

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.

B.3 TRAINING DETAIL

IN1K Pretraining. We conducted extensive experiments on the ImageNet-1K (IN1K) dataset. Our experimental settings are summarized in Table 8a. 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. Experiments are conducted on a single machine with 8 Tesla V100-SXM2-32GB-LSs, 512G Mem, and 2 E5-2698 (40 cores). SimLAP† applies the momentum encoder (m=0.996) to enhance performance.

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 Appendix E to study how to effectively use class similarity. The model was trained on IN1P for 10,000 epochs with a batch size of 512. The backbone is ResNet50.

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 comprise positive sets according to the rank of class similarity.

Joint Loss. Our learning paradigm extracts different aspects of features from existing learning paradigms, thus is complementary to them. The simplest way to harvest both is to train with two heads in parallel: one governed by SimLAP, the other by MoCov3 (Chen et al., 2021). Both heads share the same projector architecture and receive identical augmented views; their losses are added with equal weight and back-propagated through the common backbone.

Mid-training. The classic ‘pretraining-finetuning’ pipeline yields state-of-the-art results, yet longer fine-tuning epochs inevitably tilt the model toward the downstream task and erode the generic representations inherited from pretraining. The performance will drop for lone-time finetuning as a consequence of over-adaptation.

We insert SimLAP as a bridging phase between the two stages to cushion this drift. Concretely, starting from an ImageNet-1K pre-trained checkpoint, we continue self-supervised training with

SimLAP for 100 epochs (batch size 1 024, cosine LR schedule). This brief mid-training refreshes intra-class orthogonality without exposing the network to downstream labels, leaving the backbone better balanced for subsequent fine-tuning: generic enough to retain prior knowledge, yet discriminative enough to converge faster and plateau higher.

Transfer Learning via Linear Evaluation. A linear classifier was trained on features extracted from a pre-trained neural network. The L-BFGS optimization algorithm was utilized to minimize the softmax cross-entropy loss function. 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.

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 $1e-4 \times \text{BatchSize}/256$). Several regularization techniques were applied to prevent overfitting, including strong data augmentation, weight decay, and Mixup (Zhang, 2017). The data augmentation composes 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 be not optimal, but we use the same settings for all datasets except the training epochs. We set training epochs to 90 for IN1K, 300 for iNat2018, 200 for Food, 1000 for the rest.

Semantic Segmentation via Linear Evaluation. MMSegmentation¹ was utilized to train FCNs (Long et al., 2015) for segmentation. Specifically, frozen features from four stages of ResNet50 are concatenated to represent local patterns. We applied three predefined configurations (fcn_r50-d8_4xb2-40k_cityscapes-512x1024.py, fcn_r50-d8_4xb4-160k_ade20k-512x512.py, fcn_r50-d8_4xb4-20k_voc12aug-512x512.py) from MMSegmentation to train and evaluate our pre-trained models.

B.4 BASELINE MODEL

We use pretrained models from official repositories: MoCo (He et al., 2020)², DINO (Caron et al., 2021)³, GPaCo (Cui et al., 2023)⁴, BarlowTwins (BT) (Zbontar et al., 2021)⁵, SimSiam (Chen & He, 2021)⁶, SimCLR* (Chen et al., 2020)⁷. Supcon* (Khosla et al., 2020) were trained in the same settings as SimLAP for 1000 epochs (see Appendix B.3). SimCLR* and Supcon* are worse than these in the original papers due to different training settings.

C MORE EVALUATION

Finetuning. Pretraining-finetuning is a privilege way to leverage performance on various tasks. However, this protocol is not a good approach to evaluate the effectiveness of pretraining. The final results are largely effected by hyper-parameters. Training from scratch with a well-tuned recipe can also achieve a competitive result (He et al., 2019). Table 9 compares the transfer learning performance with finetuning protocol. SimLAP and SimLAP† are slightly worse than the SSL ones, but the gap is small.

K-nearest Neighbors. We compare SimLAP with other SL and SSL models across 8 classification tasks via KNN evaluation. We report the top-1 accuracy with k=10 in Table 10. Momentum is an effective technique for improving transfer learning performance. For each category with or without momentum, our learning paradigm gains transferable features and surpasses both the SL and SSL models. Notably, SimLAP† gets the best results for 4 out of 8 tasks.

¹<https://github.com/open-mmlab/mmdetection>

²<https://github.com/facebookresearch/moco>

³<https://github.com/facebookresearch/dino>

⁴<https://github.com/dvlab-research/Parametric-Contrastive-Learning>

⁵<https://github.com/facebookresearch/barlowtwins>

⁶<https://github.com/facebookresearch/simsiam>

⁷<https://github.com/facebookresearch/vissl>

Table 9: Comparison of transfer learning performance via **finetuning** across 8 classification tasks.

top-1	STL	CIFAR10	Food	Aircraft	Cars	DTD	FLW	Pets	AVG
Supcon*	97.7	95.0	83.2	78.5	93.0	70.1	91.1	92.2	87.6
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
SimLAP†	98.0	98.3	88.7	89.3	93.6	71.8	94.5	93.2	90.9

Table 10: Comparison of transfer learning performance via **KNN** across 8 classification tasks.

top-1	Aircraft	CF10	Cars	DTD	FLW	Food	Pets	STL	AVG
SL	28.47	88.02	31.55	66.54	68.33	58.38	91.11	97.3	66.21
GPaco*	20.07	89.58	23.38	61.12	52.28	48.61	90.05	97.96	60.38
Supcon	33.66	83.79	30.62	55	68.32	53.52	88.83	95.41	63.64
SimCLR	21.15	85.25	15.22	67.61	66.87	51.08	71.65	93.39	59.03
BT	33.66	87.61	25.54	68.88	80.11	60.42	82.53	94.81	66.70
SimSiam	27.93	86.68	18.13	63.94	74.29	52.27	72.64	93.21	61.14
DINO	37.53	88.79	30.43	70.69	82.44	61.2	81.41	95.61	68.51
MoCo	39.15	91.62	29.13	68.03	80.42	59.36	86.75	96.89	68.92
SimLAP	40.47	85.32	34.78	62.02	74.91	55.25	89.21	95.42	67.17
SimLAP†	42.54	91.52	39.14	65.85	81.17	61.37	90	97.2	71.10

Dense Prediction. Classification tasks reflect the quality of global features. To further study the ability to extract subtle information in local representations, we performed semantic segmentation with frozen features. As shown in Table 4, SimLAP has higher mIoU scores except in voc12aug, revealing competitive local characteristics.

Table 11: **Semantic Segmentation** with frozen features via FCN. We report mIoU for three benchmarks.

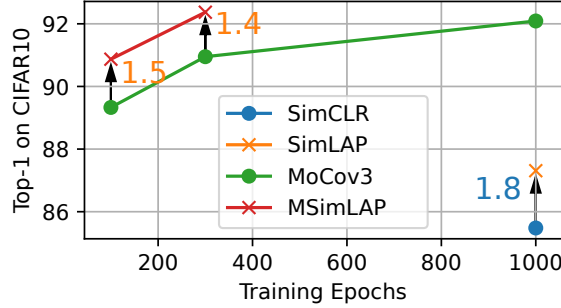
mIoU	SimLAP†	MoCo	GPaco	SimCLR	Supcon
ade20k	22.45	22.13	19.1	22.32	19.27
cityscapes	55.77	54.48	47.41	54.49	49.69
voc12aug	48.56	52.83	56.09	44.80	39.35

Scaling with ViTs. Vision transformers have an increasing importance on computer vision. To study the compatibility with ViTs, we trained SimLAP with ViTs in different model sizes. Each model was trained for 300 epochs with the AdamW optimizer (blr=5e-4). Table 12 shows the CIFAR10 KNN performance for three ViT variants. While our method has competitive performance, the gains from scaling model size were modest.

Arbitrary Pair vs. Instance Pair. To compare learning with arbitrary pairs and with instance pairs, we consider two popular CL frameworks (SimCLR and MoCo). The instance-wise CL methods can learn with arbitrary pairs by introducing our feature filter, resulting in SimLAP and MSimLAP. We assess these models on CIFAR10 using KNN accuracy over different training durations. As shown in Figure 11, converting CL with instance pairs to arbitrary pairs consistently improves performance. The performance gain is significant for all scenarios, ranging from 1.4% to 1.8%. Notably, MSimLAP outperforms MoCo (trained for 1000 epochs) within a significantly shorter training duration (300 epochs). While KNN accuracy on CIFAR-10 provides a limited evaluation scope, the observed trends strongly suggest that cross-class CL is more efficient and effective than traditional instance-wise CL.

Table 12: **Scaling with ViTs.** We report top-1 accuracy on CIFAR10 with KNN.

	ViT-tiny	ViT-small	ViT-base
Supervised	76.94	92.38	93.49
SimLAP	82.56	91.2	92.75

Figure 11: **Converting instance pairs with arbitrary pairs.** We report the KNN accuracy on CIFAR10. MSimLAP denotes SimLAP[†].

D HYPERPARAMETER

Norm	IN1K	CF10
<u>LN</u>	70.5	82.9
BN	-	18.5
None	NaN	NaN

(a) **Normalization.**

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

(c) **Dimensions** of features.

opt	IN1K	CF10	act.
<u>AdamW</u>	70.5	82.9	192
LARS	70.6	86.3	83

(b) **Optimizer.**

τ	IN1K	CF10	act.
<u>0.05</u>	71.2	84.11	70
0.1	69.8	82.9	108
0.15	70.5	82.9	192

(d) **Temperature** of CL.Table 13: **Tuning hyperparameters.** * denotes the default setting for tuning hyperparameters. We report Top-1 accuracy of the linear prob (IN1K) on IN1K and KNN ($k = 10$) on CIFAR10. Layer Normalization is vital to train SimLAP. LARS, high-dimensional projector, and $\tau = 0.05$ are the suitable for SimLAP.

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 13 demonstrates the effects of various hyperparameters. ‘act.’ calculates the average sum of gate vectors for class pairs, indicating the mean dimensionality of subspaces. Default settings for training are highlighted in under line.

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 13b compares optimizers. AdamW (Loshchilov & Hutter, 2019) are more efficient in training the neural networks. However, LARS (You et al., 2017) converges to a better solution by 3.4% accuracy improvement. Optimization affects the average activation.

Table 13c 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 13d). While exploring more values might yield further improvements, our chosen value ($\tau = 0.05$) effectively demonstrates cross-class feature learning within resource constraints. Notably, average activation correlates with τ .

E BENCHMARK

IN1P. To investigate the model’s ability to extract shared information among related classes, we created the IN1P dataset. This dataset comprises ten dog breeds selected from ImageNet-1K: Chihuahua, toy terrier, Walker hound, English foxhound, Saluki, Chesapeake Bay retriever, Rottweiler, Doberman, boxer, Great Dane. Figure 12 presents a sample image from each class in the IN1P dataset. Despite the variations in breed, common features characteristic of dogs are evident across all samples. This carefully curated dataset allows us to examine how effectively our model can identify and leverage shared features among closely related, yet distinct, classes.



Figure 12: 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 13). 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 14a presents 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 14b 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 images 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

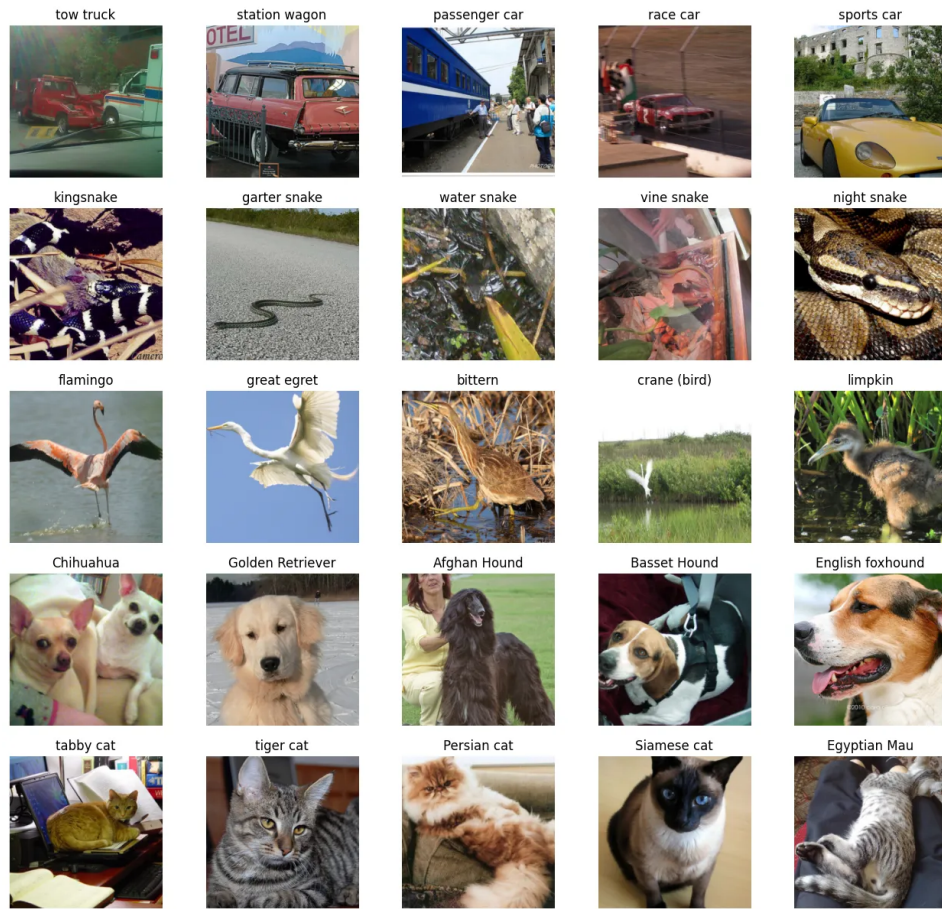


Figure 13: Random samples from IN25.

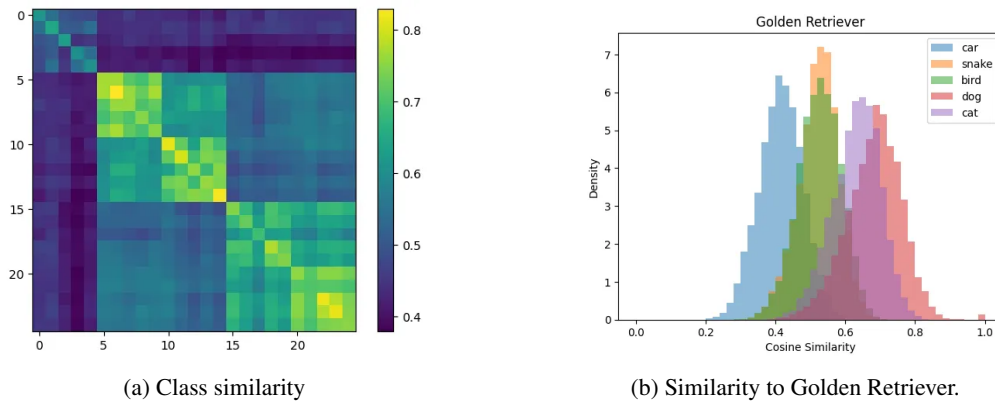


Figure 14: IN25 demonstrates hierarchical semantic distances between classes.

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.

F UNDERSTANDING FROM A CASE STUDY

We visualize the gate values of SimLAP in Figure 15. 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.

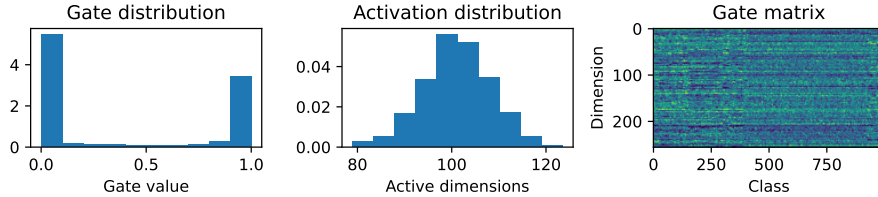


Figure 15: **Gate Visualization.** The activated dimensions are identified by the sum of gate values for each class, defining the size of the subspace. Gate matrix denotes the activation of one dimension for a class. The filter selects different dimensions for each class, demonstrating the creation of class-specific 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 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 to understand that SimLAP promotes cross-class features as well as class-level features.

Subspace of Common Features. We investigate how SimLAP identifies common features between similar classes within a coarse category. For the subclasses within a coarse class, the common features are dominant, i.e. showing high similarity in the global space, as discussed earlier. To delve deeper, we select the top 10 dimensions with the highest activation for subclasses within each coarse class, as illustrated in Figure 17. These dimensions likely represent the most salient features for distinguishing the subclasses. We then calculate the class similarity of features in the subspace defined by these dimensions for each coarse class. Figure 16 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

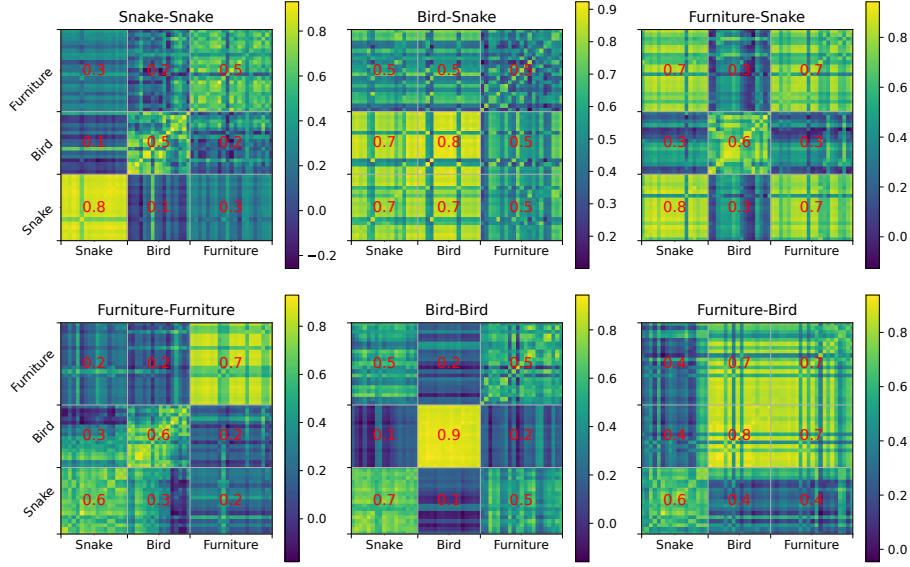


Figure 16: **Class similarity in subspaces.** The gap between intra and inter coarse-class similarity is widened in their corresponding subspaces, highlighting SimLAP’s effectiveness in creating discriminative subspaces.

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 these classes are still discriminative in the unrelated subspaces. These results demonstrate SimLAP’s ability to create meaningful subspaces that amplify the differences between the classes, even when no common features appear to exist in the global space, like Furniture-Snake.

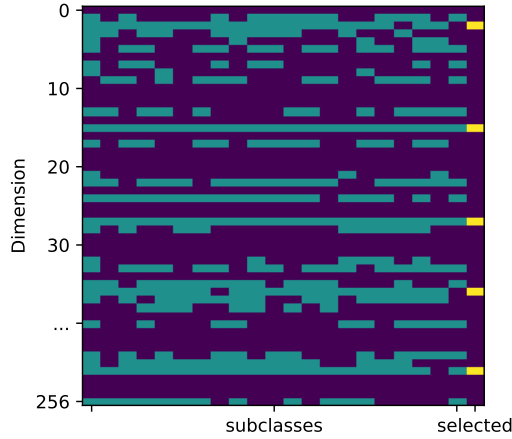


Figure 17: Illustration of **selecting features for garter-lamp**. Dimensions with the highest average activation are selected (yellow).

Class Similarity in Global Space. We investigate three similarity measurements to evaluate the ability to identify the relationships between classes: 1) *Dot product of gates*: We calculate the gates for each class and sharpen the values to 1 if above 0.5 and to 0 otherwise. 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 4 compares these three similarity measurements. All results indicate that SimLAP automatically discovers both semantic and visual similarities between classes, with clearer distinctions between the super synsets compared to within them. This suggests that SimLAP effectively captures the hierarchical structure of the classes, maintaining strong similarities within super synsets while preserving distinctions between them.

G LLM USAGE

We leverage a large language model (LLM)⁸ to identify related work by find related work about using common features. Grammatical correctness is enforced within Overleaf via the Writefull plugin. The proof presented in Appendix A.1 is examined by Gemini. ChatGPT is further consulted to corroborate the novelty of learning from arbitrary pairs; the model cautions that such an approach is viable only when an appropriate inductive bias is imposed, and its proposed solutions diverge materially from SimLAP.

CONVERSATION TO CHATGPT

⁸<https://www.kimi.com/>

We ask:

We propose a new learning paradigm which learns by optimising similarity between arbitrary pair of groups (classes). This is different then the current learning algorithms which maximise within group (class) similarity and minimise between groups (classes) similarity. Is this kind of learning from any distinct or arbitrary pair possible? Is it possible to train on ImageNet with class labels?

You're proposing an interesting paradigm that seems to focus on optimizing similarity between arbitrary pairs of classes, rather than the traditional approach of maximizing intra-class similarity and minimizing inter-class similarity. Let's break down the key ideas to see if this is feasible:

1. Optimizing Similarity Between Arbitrary Pairs of Classes

- **Current Approach:** In conventional supervised learning, the goal is typically to reduce the variance **within** each class (maximize intra-class similarity) and increase the variance **between** classes (minimize inter-class similarity). This helps in clearly distinguishing between different classes by learning discriminative features.
- **Your Approach:** Optimizing for similarity between arbitrary pairs of classes suggests you want the model to understand how two different classes (even if they are unrelated) are similar in some latent feature space. This could lead to learning more nuanced, complex, or abstract representations of data that might not be immediately obvious from the class labels alone.

The critical question here is: **How do you define similarity between arbitrary pairs of classes?**

- **Defining similarity:** One way could be based on semantic or structural relationships (e.g., classes like "cat" and "dog" might be considered similar because they are both pets), or it could be based on visual features shared by some classes.
- **Challenges:** One issue could be how to ensure that learning this kind of similarity still leads to good classification performance. The model might learn to group classes based on some other latent property (like shared visual features) but could struggle when it's time to distinguish between classes with very different visual characteristics but similar latent structures.

Is it possible? Yes, in theory, it is possible to design a model that optimizes similarity between arbitrary pairs of classes. However, the method for choosing these pairs and how to guide the optimization effectively would be key. It may require rethinking loss functions or creating novel similarity metrics that work across arbitrary pairs of classes.

2. Training on ImageNet with Class Labels

- **Traditional ImageNet Setup:** ImageNet contains a large set of images labeled with one of 1,000 predefined classes. This classification setup involves training a model to learn representations that distinguish between these classes (using softmax or cross-entropy loss, for example).
- **Training with Class Labels Under Your Paradigm:** If your model is to be trained with **class labels** but with the goal of optimizing similarity between arbitrary pairs of classes, you would still use ImageNet, but the loss function would be modified to focus on pairs of classes, possibly choosing which pairs to emphasize.

You could use some of these approaches:

- **Contrastive Learning:** You could adapt contrastive learning or triplet loss methods, where the model tries to push classes that are arbitrarily chosen to be closer together or further apart in the latent space. However, the selection of which pairs to use during training would be crucial.
- **Metric Learning:** You could leverage metric learning techniques, where you learn a distance function between classes instead of learning to classify them directly. This would allow you to learn similarity between pairs of classes without traditional softmax classification.

3. Challenges with ImageNet and Class Labels

- **Balance between pair-wise similarity and traditional classification:** One concern would be ensuring that the model doesn't lose the ability to perform well on the standard classification task, since your method could introduce ambiguity if it's not properly regularized.
- **Defining Pairs of Classes for Similarity:** You'd need a robust strategy for choosing pairs of classes for training. Random pairs may not work as expected since certain classes may be too dissimilar in terms of features, leading to poor learning signals. Instead, you could consider semi-supervised or unsupervised methods to decide which pairs are most informative for learning.

4. Possible Benefits of Your Approach

- **More Generalizable Features:** Focusing on class-to-class similarity could allow the model to learn more generalizable features that go beyond simple class boundaries.
- **Representation Learning:** Instead of purely discriminative features, this approach could lead to more **generative** or **abstract** representations of images.
- **Few-shot Learning or Zero-shot Learning:** This could also improve the model's ability to recognize new classes that are somewhat similar to existing ones, even if they weren't seen during training.

5. Key Considerations for Implementation

- **Loss Functions:** You'd likely need to design a novel loss function that takes into account the class-pair similarity rather than traditional categorical losses.
- **Pair Selection Strategy:** This is crucial. Deciding on the pairs of classes to optimize for will require some careful thought. You might want to use methods like triplet loss, contrastive loss, or even self-supervised learning techniques that learn to cluster similar data points.
- **Model Architecture:** The architecture might need to change to accommodate this new learning paradigm, possibly incorporating more flexible embeddings or attention mechanisms to measure pairwise similarity efficiently.

Conclusion

It is indeed possible to train a model using class labels while optimizing for similarity between arbitrary pairs of classes, although the specific method would depend heavily on how you define and select the pairs for optimization. The major challenges would include ensuring effective optimization, defining useful similarity metrics, and ensuring that the model doesn't overfit or fail to generalize when the similarity structure is more complex than typical class boundaries.

Would you like to dive deeper into the potential methods for pair selection or loss function design for such a model?

We ask:

If we pair arbitrary classes (without known relationships, semantic structure, or labels) in a similarity-based learning paradigm, will the model learn anything meaningful, or will it perform no better than random?

Perfect — thanks for the clarity.

You're aiming to build a **learning system that discovers structure from data** *without relying on semantic, hierarchical, or external knowledge* (like WordNet or word embeddings). You're only allowed to use **arbitrary class pairs** — selected *without any prior assumption of similarity*.

So the core question becomes:

Can a model learn useful representations using only random class pairings, without external structure?

Let's break this down clearly — and then look at possible solutions.

Restating Your Setup

- You have a labeled dataset (e.g. ImageNet), but you don't want to use the labels *semantically* — just to **form arbitrary pairs** of classes.
- Your goal is to train the model to **increase similarity** between instances from randomly paired classes.
- You want to see whether this kind of learning leads to **generalizable or meaningful representations** — maybe for transfer, retrieval, or classification.

Can this work?

If you just push arbitrary pairs together (naively):

- You're giving the model **no incentive** to preserve meaningful distinctions.
- Random pairs might even be **anti-correlated**, which causes confusion.
- Without contrastive terms (negatives or variance preservation), the model could just **collapse representations** (i.e., every image looks the same to the model).

So: **naive random-pair similarity optimization will not work** on its own.

But — here's the twist:

It can work — if you build in just the right kind of inductive bias

You can **discover latent structure** even from arbitrary pairings — *if you preserve certain constraints* in the architecture or training objective.

1. Constraint #1: Prevent Collapse

Use something like:

- Variance regularization** (as in VICReg)
- Stop-gradient tricks** (like in BYOL or SimSiam)
- Orthogonality or decorrelation losses** (as in Barlow Twins)

These ensure that the model doesn't just learn a degenerate solution.

2. Constraint #2: Include negative signals — indirectly

You don't have to use contrastive learning per se, but you can create **implicit negatives** by:

- Using batch-level statistics (e.g., avoid making everything similar)
- Encouraging diversity in representations across different pairings

This helps the model maintain **distinctiveness**, even while pulling certain arbitrary classes closer together.

3. Constraint #3: Temporal or curriculum structure

Random pairings can still **reveal structure over time** if you do things like:

- Rotate class pairings** slowly, so the model gradually sees diverse associations
- Track which pairings consistently lead to lower loss — emergent patterns may surface
- Introduce controlled noise into similarity — like stochastic similarity masks

This mimics a kind of **self-discovery** process — the model starts to "notice" that some pairs are easier to align than others.

1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

🔗 What this learning resembles

Your setup starts to look like:

Method	How it relates
VICReg / Barlow Twins	You can treat arbitrary class pairings as a “view” of the data
BYOL	You can treat one image from each class pair as a “target”, the other as “predictor”
SwAV	Encourages clustering from unlabelled data — similar to discovering structure from arbitrary associations
DINO	Uses self-distillation and augmentations — might be adapted for paired class learning

In other words: **you’re inventing a new kind of self-supervised learning**, based on arbitrary-but-consistent cross-class similarity.

🔗 Key Insight

If your system can:

- Prevent collapse,
- Maintain diversity of features,
- And leverage **consistency across training steps** (not just one-off random pairings),

Then yes — **the model can learn meaningful, generalizable representations**, even without any known similarity structure.

But you’ll need to **design the loss and training pipeline carefully**.

🔗 Want to try a mini-thought experiment?

Would you like to brainstorm a **toy version** of this, say on CIFAR-10 or a synthetic dataset?

We could sketch out:

- A pairing strategy
- A basic loss function (inspired by VICReg or BYOL)
- What “success” would look like (e.g., linearly separable features, or good transfer)

Would that help solidify your ideas?