LiDAR: Sensing Linear Probing Performance in Joint Embedding SSL Architectures

Vimal Thilak [*] **Chen Huang** **Omid Saremi** **Laurent Dinh**
Hanlin Goh **Preetum Nakkiran** **Josh Susskind** **Etai Littwin**
Apple

Abstract

Joint embedding (JE) architectures have emerged as a promising avenue for acquiring transferable data representations. A key obstacle to using JE methods, however, is the inherent challenge of evaluating learned representations without access to a downstream task, and an annotated dataset. Without efficient and reliable evaluation, it is difficult to iterate on architectural and training choices for JE methods. In this paper, we introduce *LiDAR* (Linear Discriminant Analysis Rank), a metric designed to measure the quality of representations within JE architectures. Our metric addresses several shortcomings of recent approaches based on feature covariance rank by discriminating between informative and uninformative features. In essence, *LiDAR* quantifies the rank of the Linear Discriminant Analysis (LDA) matrix associated with the surrogate SSL task—a measure that intuitively captures the information content as it pertains to solving the SSL task. We empirically demonstrate that *LiDAR* significantly surpasses naive rank based approaches in its predictive power of optimal hyperparameters. Our proposed criterion presents a more robust and intuitive means of assessing the quality of representations within JE architectures, which we hope facilitates broader adoption of these powerful techniques in various domains.

1 Introduction

In recent years, self-supervised learning (SSL) has emerged as a pivotal technique for pretraining representations on extensive, unlabeled datasets, thus effectively alleviating the often burdensome labeling requirements (Chen et al., 2020; Assran et al., 2023; Chen & He, 2020; Caron et al., 2021; Bardes et al., 2021; Caron et al., 2018, 2020; Baevski et al., 2022; Zbontar et al., 2021; He et al., 2021; HaoChen et al., 2021; Grill et al., 2020). However, despite the remarkable progress made in SSL, assessing the quality of representations acquired through this method remains an open problem. This challenge is further amplified when considering *Joint Embedding* (JE) architectures, which notoriously suffer from uninterpretable loss curves that offer little clue to assess the progression of training. The conventional and widely adopted approach involves evaluating model performance by employing these representations in downstream tasks. Nevertheless, when the objective is to learn versatile representations applicable across diverse domains, this method demands substantial investments of time and resources to comprehensively evaluate performance across a multitude of tasks and datasets. Alternatively, limiting assessments to only a few datasets and tasks undermines confidence in the evaluation process.

As a result, a fundamental question arises: Can we effectively evaluate the quality of learned representations without relying on explicit downstream task evaluations? Addressing this inquiry necessitates the precise definition of ”quality” in the context of representations. Subsequently, we must explore statistical estimators capable of quantifying this quality without depending on downstream

[*]Corresponding Authors: {vthilak, jsusskind, elittwin}@apple.com

evaluations. Beyond its theoretical implications, such a metric holds significant practical value, as it aids in model selection and the development of novel SSL algorithms.

Recent literature has introduced metrics that share a common foundation, relying on statistics derived from the empirical covariance matrices taken at different layers of the model (Garrido et al., 2022; Agrawal et al., 2022b). Notably, the recently introduced *RankMe* (Garrido et al., 2022) method shows that the rank of the embeddings (closely related to the rank of the feature covariance) correlates surprisingly well with downstream performance, demonstrating SOTA results in label free hyperparameter selection. W refer the interested reader to Appendix A for the definition and details of RankMe. In this paper, we build upon *RankMe* by proposing a simple modification to the feature covariance matrix on which the rank is calculated, which significantly and consistently improves its predictive power of downstream performance. Our method, which we refer to as *LiDAR*, is motivated by a simple observation: the covariance spectrum can be easily and arbitrarily manipulated, and this manipulation is often encouraged by implicit or explicit regularizations in the SSL objectives. Consequently, a representation with a full rank covariance matrix may result from spurious factors rather than representing rich semantic features. As a result, even though methods such as *RankMe* demonstrate impressive results, we show that non-trivial additional gains can be had rather effortlessly. We summarize our contributions in the following: **a)** We introduce *LiDAR*, a method for assessing representation quality of JE SSL objectives, and theoretically motivate it. *LiDAR* uses the SSL objective in its definition, providing a more intuitive and robust metric for assessing SSL representations. **b)** We conduct a comprehensive set of experiments spanning multiple JE architectures, both Transformer (Dosovitskiy et al., 2021) and ResNet He et al. (2016) based backbones. These include contrastive and regularized methods, as well as newer models such as I-JEPA (Assran et al., 2023) and data2vec (Baevski et al., 2022), which leverage masking techniques. We demonstrate that the *LiDAR* metric correlates significantly and consistently higher with downstream linear probing performance than *RankMe* as measured by both the Spearman Rank and Kendall rank correlation coefficient or Kendall's τ. **c)** We show that *LiDAR* demonstrates consistently strong performance in hyperparameter selection, outperforming *RankMe*. We further demonstrate this capability in randomized trials, where multiple hyperparameters are varied simultaneously.

2 Method

Our method is based on *linear discriminant analysis* (Bishop, 2007), a classical label-aware dimensionality reduction and classification algorithm which we adopt to the multi-view setting common in the SSL literature. Absent downstream task and labels, we use clean samples as surrogate classes. For an input distribution $\mathcal{D}$, consider a generic JE architecture with an embedding function e. For any clean input x, let $\mathcal{D}_x$ denote a conditional distribution over all transformed [2] inputs given x. Define $\mu_x = \mathbb{E}_{\tilde{x}\sim\mathcal{D}_x}[e(\tilde{x})]$ and $\mu = \mathbb{E}_{x\sim\mathcal{D}}[\mu_x]$. we define the generalized covariance Σ_{lidar} by:

$$\Sigma_{\text{lidar}}(e) = \Sigma_{\text{w}}(e)^{-\frac{1}{2}}\Sigma_b(e)\Sigma_{\text{w}}(e)^{-\frac{1}{2}} \tag{1}$$

where $\Sigma_b(e) = \mathbb{E}_{x\sim\mathcal{D}}\left[(\mu_x - \mu)(\mu_x - \mu)^\top\right]$, $\Sigma_{\text{w}}(e) = \mathbb{E}_{x\sim\mathcal{D}}\mathbb{E}_{\tilde{x}\sim\mathcal{D}_x}\left[(e(\tilde{x}) - \mu_x)(e(\tilde{x}) - \mu_x)^\top\right] + \delta I_p$ and δ is a small positive constant, and I_p is the identity matrix of dimension p. Let $\lambda = \lambda_1, ..., \lambda_p$ be the eigenvalues of Σ_{lidar}, then *LiDAR* is defined by applying the smooth rank measure introduced in (Roy & Vetterli, 2007) on Σ_{lidar} :

$$\textit{LiDAR}(e) = \exp\Big(-\sum_i p_i \log p_i\Big), \quad p_i = \ (\ \lambda_i/\|\lambda\|_1) + \epsilon \tag{2}$$

where ϵ is a small positive constant. In practice, we use unbiased estimates of Σ_w, Σ_b using hyperparameters n, q where n is the numbers of surrogate classes (clean samples), and q is the number of transformed samples per class. Note that, as in classical LDA, the eigenvalues $\lambda_1, ..., \lambda_p$ measure variance along discriminative directions. Hence, *LiDAR*(e) takes into account the SSL objective that is being optimized, and ignores directions of variability in e that are useless in solving the SSL task. We postulate the existence of such directions, as JE techniques implicitly or explicitly incorporate measures to preserve representations from collapsing. In simpler terms, some variations in e may not necessarily arise from high-level semantic features driven by the SSL objective, but rather from arbitrary attributes influenced by regularization. This phenomenon can clearly be seen in I-JEPA and

[2] Data augmentations, or otherwise data points which are treated as positive samples

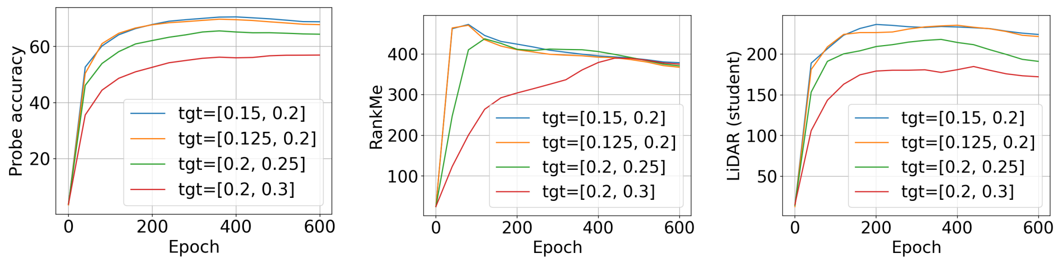


Figure 1: ViT-Base architecture trained with I-JEPA (Assran et al., 2023) by varying the target mask scale hyperparametr. We observe that (1) *RankMe* correlates poorly with downstream performance for most models, with the exception of the worst performing model, and (2) *LiDAR* correlates highly with downstream performance for all models.

Table 1: I-JEPA: Kendall's τ coefficient between effective ranks of RankMe, RankMe (aug.) and LiDAR and linear probe accuracy. Hyperparameters include learning rate, weight decay, target and context mask scales varied via grid or random search. Table 9 shows a breakdown for grid search.

Hyperparameter Search	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Grid	0.5830	0.7513	**0.8159**	0.5713	0.6828
Random	0.8314	**0.8989**	0.8434	0.8487	0.8616

data2vec, where the rank of a representation inflates early on in training, only to rapidly diminish, with peak downstream performance occurring far from the point of maximal rank. This makes any standard data covariance rank based measures extremely limited in their ability to spot the point of optimal performance in a training run. In contrast, *LiDAR* initiates at a much lower rank and steadily ascends, more closely aligning with downstream linear probing performance, as depicted in Figure 1 and Figure 14. We delve deeper into the theoretical rationale behind *LiDAR* in Appendix B.

3 Experimental Results

We present experimental results that show that he *LiDAR* metric correlates surprisingly well with downstream performance, as measured by linear probing. In the vast majority of experiments, we see a significant improvement over *RankMe* in terms of the Kendall's τ and Spearman's rank correlation to the oracle, and an improved performance in hyperparameter selection. Due to space constraints, we defer the reader to Appendix C and Appendix E for a comprehensive view of experimental details, supplementary empirical results, and additional figures. In the main text, we have included a representative selection to provide an overview.

Tables 1 and 2 present a comprehensive analysis of the results obtained for the I-JEPA. The evaluation involves a comparison between *LiDAR*, assessed on both the teacher and student branches, and *RankMe*, with and without data augmentation, alongside the oracle reference. We observe a general trend where *LiDAR* applied on the student branch correlates much higher with the oracle than the teacher branch. Overall we observe significant outperformance of *LiDAR* over *RankMe* applied on clean and augmented inputs for all hyperparameters tested. A noteworthy observation from Table 2 is that *RankMe*, when coupled with data augmentation to compute the feature covariance matrix,

Table 2: I-JEPA: Linear probe accuracy recovered by *RankMe* and *LiDAR* on ImageNet-1K dataset. Hyperparameters set via grid search. Table 10 shows detailed breakdown for grid search.

Hyperparameter Search	ImageNet Oracle	RankMe (Student)	RankMe (aug.) (Student)	LiDAR (Teacher)	RankMe (aug.) (Teacher)	LiDAR
Grid	70.5800	59.8960	**67.8680**	**67.8680**	59.8960	67.5420
Random	68.9840	53.3700	53.3700	61.1760	53.3700	**66.8880**

can match *LiDAR*'s performance for the best-selected model, albeit trailing significantly when data augmentation is not employed. Table 1 and Table 2 present results from randomized trials where hyperparameters are randomly sampled within predefined ranges. In these trials, a consistent trend of *LiDAR* outperforming both versions of *RankMe* is evident, as can also be seen in Figure 2. Tables 13a and 13b in the appendix extend our analysis to the data2vec model, and the findings parallel those observed for I-JEPA. Notably, *LiDAR* consistently selects more performant hyperparameters than both variants of *RankMe*, narrowly missing oracle-level performance.

Table 3 summarizes our results for VICReg (Bardes et al., 2021) and SimCLR (Chen et al., 2020). This table reports metrics calculated at at the end of training process, i.e., after 100 epochs of training. We observe significantly higher correlations with the oracle performance. Performance over additional checkpoints are presented in Table 7. It's worth highlighting that, with VICReg, *LiDAR* achieves optimal results when applied to the representation rather than the embedding, as embedding-based evaluations result in dramatic performance degradation, a phenomenon aligning with the non-monotonic relationship between rank and performance reported by (Garrido et al., 2022). This illustrates that high rank is a necessary but not a sufficient condition for high performance. Further validation of *LiDAR*'s efficacy in hyperparameter selection is provided in Appendix Table 5, where the method achieves results that are within a fraction of a percentage point from oracle-level performance across all considered hyperparameters.

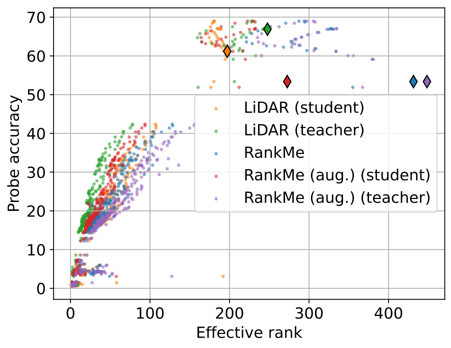


Figure 2: I-JEPA: ViT-Base architecture trained on Imagenet-1K with 20 sets of hyperparameters drawn uniformly at random. Each point represents a checkpoint while the ◇ marker represents the model selected by each metric. We observe that (1) *LiDAR* outperforms *RankMe* (Table 1), (2) augmented-*RankMe*, our variant of *RankMe*, shows the highest correlation (Table 1), and (3) remarkably *LiDAR* outperforms the other methods in selecting hyperparameters by recovering the highest downstream performance.

Table 3 also reports results for SimCLR, a widely used contrastive self-supervised learning method. The evaluation centers on the final checkpoint obtained after 100 training epochs. Table 3a reveals that *LiDAR* consistently demonstrates the highest correlation among the three metrics under scrutiny. Moreover, Table 3b confirms that *LiDAR* consistently selects the most optimal hyperparameters among the three methods considered. Lastly, Table 17 lists the results for a ViT-Small trained with DINO (Caron et al., 2021) on ImageNet-1K dataset. We observe that *LiDAR* evaluated with both the teacher and student branches show stronger correlation than *RankMe*.

4 Conclusion

We introduce *LiDAR*, a novel approach to evaluate self-supervised learning models, which builds upon the foundation laid by *RankMe*. We experimentally demonstrate *LiDAR*'s utility over covariance-rank-based approaches, enabling accurate and label-independent assessment of learned representations in SSL. Our method provides a powerful tool for practitioners seeking to efficiently optimize their models in data-scarce environments, and we hope it can be integrated into the standard toolkit of model evaluation in self-supervised learning.

Table 3: SimCLR and VICReg: Performance of *RankMe*, augmented-*RankMe* and *LiDAR* on ImageNet-1K dataset at the end of training.

(a) Kendall's τ correlation coefficient.

SSL Method	RankMe	RankMe (aug. dataset)	LiDAR
SimCLR	0.4982	0.5761	**0.8167**
VICReg	0.2056	0.3790	**0.8105**

(b) Linear probe accuracy.

SSL Method	ImageNet Oracle	RankMe	RankMe (aug. dataset)	LiDAR
SimCLR	59.14	56.46	57.83	**58.93**
VICReg	64.74	63.95	63.95	**64.71**

References

Kumar Krishna Agrawal, Arnab Kumar Mondal, Arna Ghosh, and Blake Aaron Richards. $\$\backslash\alpha\$$-req : Assessing representation quality in self-supervised learning by measuring eigenspectrum decay. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022a. URL `https://openreview.net/forum?id=ii9X4vtZGTZ`.

Kumar Krishna Agrawal, Arnab Kumar Mondal, Arna Ghosh, and Blake Aaron Richards. α-req : Assessing representation quality in self-supervised learning by measuring eigenspectrum decay. In *Neural Information Processing Systems*, 2022b. URL `https://api.semanticscholar.org/CorpusID:258509089`.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael G. Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023. URL `https://api.semanticscholar.org/CorpusID:255999752`.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning*, 2022. URL `https://api.semanticscholar.org/CorpusID:246652264`.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *ArXiv*, abs/2105.04906, 2021. URL `https://api.semanticscholar.org/CorpusID:234357520`.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1 edition, 2007. ISBN 0387310738. URL `http://www.amazon.com/Pattern-Recognition-Learning-Information-Statistics/dp/0387310738%3FSubscriptionId%3D13CT5CVB80YFWJEPWS02%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0387310738`.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018. URL `https://api.semanticscholar.org/CorpusID:49865868`.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *ArXiv*, abs/2006.09882, 2020. URL `https://api.semanticscholar.org/CorpusID:219721240`.

Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9630–9640, 2021. URL `https://api.semanticscholar.org/CorpusID:233444273`.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *ArXiv*, abs/2002.05709, 2020. URL `https://api.semanticscholar.org/CorpusID:211096730`.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15745–15753, 2020. URL `https://api.semanticscholar.org/CorpusID:227118869`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann LeCun. Rankme: Assessing the downstream performance of pretrained self-supervised representations by their rank. *ArXiv*, abs/2210.02885, 2022. URL `https://api.semanticscholar.org/CorpusID:252735059`.

Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020. URL `https://api.semanticscholar.org/CorpusID:219687798`.

Jeff Z. HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. In *Neural Information Processing Systems*, 2021. URL `https://api.semanticscholar.org/CorpusID:235367888`.

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, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Doll'ar, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2021. URL `https://api.semanticscholar.org/CorpusID:243985980`.

Tianyu Hua, Wenxiao Wang, Zihui Xue, Yue Wang, Sucheng Ren, and Hang Zhao. On feature decorrelation in self-supervised learning. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9578–9588, 2021. URL `https://api.semanticscholar.org/CorpusID:233481690`.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. *ArXiv*, abs/2110.09348, 2021. URL `https://api.semanticscholar.org/CorpusID:239016966`.

M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938a. ISSN 00063444. URL `http://www.jstor.org/stable/2332226`.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938b.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

Alexander C. Li, Alexei A. Efros, and Deepak Pathak. Understanding collapse in non-contrastive siamese representation learning. In *European Conference on Computer Vision*, 2022. URL `https://api.semanticscholar.org/CorpusID:252596511`.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. *2007 15th European Signal Processing Conference*, pp. 606–610, 2007. URL `https://api.semanticscholar.org/CorpusID:12184201`.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

C. Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 100(3/4):441–471, 1987. ISSN 00029556. URL `http://www.jstor.org/stable/1422689`.

Charles Spearman. The proof and measurement of association between two things. 1961.

Carsen Stringer, Marius Pachitariu, Nicholas A. Steinmetz, Matteo Carandini, and Kenneth D. Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571:361 – 365, 2019. URL `https://api.semanticscholar.org/CorpusID:256769116`.

Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks, 2017.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *ArXiv*, abs/2103.03230, 2021. URL `https://api.semanticscholar.org/CorpusID:232110471`.

Liu Ziyin, Ekdeep Singh Lubana, Masakuni Ueda, and Hidenori Tanaka. What shapes the loss landscape of self-supervised learning? *ArXiv*, abs/2210.00638, 2022. URL `https://api.semanticscholar.org/CorpusID:252683253`.

A Preliminaries

A.1 Self Supervised Learning

The main goal of self supervised pre-training is to learn a general purpose data representation $f(x)$ that is transferable to a large variety of downstream tasks. Informally, the degree of transferability of f can be measured by how easy it is to learn a downstream task given f. In practice, a set of downstream tasks $\{T_j\}$ are used to asses the transferability of f by training additional readout networks ϕ_j on top of f. That is, a successful pre-training scheme involves finding f such that there exists "simple" functions ϕ_j, such that $\phi_j \circ f$ solves T_j to a reasonable degree, and ϕ_j are "easy" to learn [3]. In the popular linear probing protocol, a linear readout functions ϕ_j is used to assess the quality of f. For the remainder of the paper we restrict our investigations to linear probing.

A.2 Joint Embedding Architectures

A common paradigm in the current practice of SSL is the multi-view setting. In its simple form, a pair of inputs are processed by a JE architecture, where each input is encoded separately by a (possibly shared) encoder. The SSL objective is then tasked with learning a representation such that "compatible" pairs that share semantic information are easily predictive of the each other, perhaps with the help of a latent variable. Various methods in the literature fall under this general category, which mostly differ by how compatible inputs are sampled, and how they avoid representational collapse. We can formally define a JE architecture with an encoder [4] $f(x) : \mathcal{X} \to \mathbb{R}^d$ and a projector function [5] $\psi(f) : \mathbb{R}^d \to \mathbb{R}^p$. We adopt the terminology in (Garrido et al., 2022), and refer to the output of the encoder f as a *representation*, and the output of the composed encoder and projector $e = \psi \circ f$ as an *embedding*. A common theme among most JE SSL objectives is that they seek to learn embedding functions that produce similar embeddings for compatible views, which are typically generated by various forms of data augmentation. However, more recent JE architecture somewhat depart from this paradigm by using input masking, and introducing latent variables in the projector function ψ. For example, in I-JEPA (Assran et al., 2023) and data2vec (Baevski et al., 2022), $\tilde{x}$ and x represent partially masked and unmasked inputs respectively, and $\psi(f; z)$ is tasked with predicting the parts of the representation $f(x)$ given spatial locations provided by z. For simplicity and without loss of generality, we remove the explicit notation z in the definition of ψ. Note that, absent any input reconstruction loss, JE architectures potentially promote more abstract representations by filtering fine grained pixel level information. However, without an input reconstruction term, the loss used is often uninformative of the actual metric of interest, which is the zero or few shot transfer of the learned representation to downstream tasks. Compounding this challenge, extended training durations can significantly deteriorate representation quality, even when employing a set of hyperparameters with proven performance, as illustrated in Figure 1. In such cases, finding an appropriate early stopping criterion is critical.

Dimensional Collapse JE architectures are prone to various forms of representation dimension collapse, which can manifest in different flavors (Ziyin et al., 2022; Jing et al., 2021; Hua et al., 2021). In contrastive methods, dimensional collapse occurs when learning results in an excessively low dimensional representations in a way that hinders downstream performance. Regularized methods, when insufficiently regularized either implicitly or explicitly, can theoretically suffer complete collapse, where the learned representation trivializes to a constant. However, beyond such a rough categorization of the training process to partial or complete collapse, recent literature alludes to a more nuanced observation that, in some settings, the feature covariance eigenspectrum can be used to accurately gauge the quality of the learned representation as it pertains to downstream performance.

A.2.1 Embeddings and Feature Covariance Eigenspectrum

Recent developments in the evaluation of JE-SSL representations have introduced methods that leverage information derived from the spectrum of either the representations or the embeddings

[3] We use the terms "simple" and "easy to learn" here loosely as requiring few samples

[4] The encoder function need not be shared between views, however for simplicity and without loss of generality we assume it is

[5] Some SSL methods do not use a projector function, in which we can assume it is the identity function

(RankMe, Garrido et al. (2022) or from the spectrum of the covariance matrix, α-Req, Agrawal et al. (2022a)). Garrido et al. (2022) who propose a new measure named *RankMe* and demonstrate that this measure, when applied across different hyperparameter configurations, exhibits a strong correlation with downstream performance across various settings and tasks. In order to define *RankMe*, we start with a dataset $\{x_i\}_{i=1}^n$ drawn iid from some input distribution $\mathcal{D}$, and an embedding function $e(x) \in \mathbb{R}^d$ that produces an embedding matrix $\mathbf{Z}$ with dimensions (n, d) whose singular values are denoted as $\sigma = \sigma_1, \sigma_2, ...\sigma_{\min(n,d)}$. These singular values reveal insights into the nature of the mapping function $f(x)$. RankMe (Garrido et al., 2022) introduced a soft measure of effective rank expressed as $\exp(-\sum_{i=1}^p p_i \log p_i)$, where $p_i = \frac{\sigma_i}{\|\sigma\|_1} + \epsilon$, and ϵ represents a small constant.

In a related work, α-Req (Agrawal et al., 2022a) uses insights from infinite dimensional spaces to argue that the eigenspectrum of representations, i.e., eigenspectrum of the feature covariance should decay at an ideal rate of $\lambda_i \sim O(i^{-1})$ where λ_i is the i^{th} eigenvalue of the covariance matrix. However, it's essential to note that any condition imposed on the eigenspectrum alone may not be sufficient to ascertain the representational power of the mapping function f. This limitation arises from the fact that a simple linear mapping, such as $f(x) = Wx$ with a weight matrix W, can manipulate the covariance matrix's spectrum arbitrarily. Additionally, even a random mapping could exhibit a high effective rank without necessarily translating into significant downstream performance improvements. Notably, (Li et al., 2022) showed that the loss value and covariance spectrum can be used in tandem to predict performance. However, their method requires training a classifier on offline data, making it highly inefficient as an unsupervised method. *LiDAR* follows a similar intuition with a more efficient and effective formulation: A representation with a high effective rank, when coupled with a low objective loss, points to successful training. To balance the two terms we leverage a discriminative method from classical statistics, repurposed to SSL settings.

B Theoretical Motivation

In this section we illustrate a heuristic as to why we might expect *LiDAR* to outperform previous methods relying on the covariance spectrum. As an exemplary JE SSL method, we consider the VICreg objective, which is comprised of three terms:

$$\mathcal{L}_{\text{VIRreg}} = \underbrace{\lambda\mathcal{L}_{\text{inv}}}_{\text{Invariance}} + \underbrace{\mu\mathcal{L}_{\text{var}} + \nu\mathcal{L}_{\text{cov}}}_{\text{Regularization}} \tag{3}$$

where λ, μ, ν are hyperparameters, and $\mathcal{L}_{\text{inv}}, \mathcal{L}_{\text{var}}, \mathcal{L}_{\text{cov}}$ are the invariance, variance and covariance terms computed over a minibatch. In this objective, the regularization term which is comprised of the variance and covariance terms explicitly encourages the embedding function's covariance to be of high rank, while the invariance term insures that compatible pairs are mapped to similar embeddings. We highlight that a low regularization loss is achievable by random embeddings, which might be high rank, but devoid of any utility as for downstream tasks. A measure of representation quality which is based on covariance rank alone would therefore, theoretically, fail to capture this failure case, as it only pertains to the regularization term. Indeed, balancing the hyperparameters λ, μ, ν is necessary to prevent superficially inflating the covariance rank. On the other hand, *LiDAR* is invariant to information that is not used to discriminate between surrogate classes in the SSL objective. To make this point concrete, consider a (centered) embedding function $e(x) : \mathbb{R}^d \to \mathbb{R}^p$, a random independent noise vector $\mu \in \mathbb{R}^r$ such that $\mathbb{E}[\mu] = \mathbf{0}, \mathbb{E}[\mu\mu^\top] = \Sigma_\mu \in \mathbb{R}^{r\times r}$, and consider the (random) embedding functions $\tilde{e}(x) = [e(x)^\top, \mu^\top]^\top : \mathbb{R}^d \to \mathbb{R}^{p+r}$. Naturally, when measuring the downstream performance of $e, \tilde{e}$, we expect that e should not be worse than $\tilde{e}$. (in practice performance is measure on the representation f, however for the sake of a clean argument we will ignore this technicality). In turn, this should, ideally, translate to *LiDAR*$(\tilde{e}) \leq$ *LiDAR*(e). *LiDAR* (unlike covariance spectrum based approaches) indeed captures this property, as illustrated in the following proposition:

Proposition 1. *Let $\mathcal{D}$ denote a distribution over inputs $x \in \mathbb{R}^d$, let $\mathcal{D}_x$ denote a conditional distribution of transformed inputs given x. Let $\lambda = \lambda_1, ..., \lambda_p$ be the eigenvalues of $\Sigma_{lidar}(e)$. Assume that $\frac{\|\lambda\|_\infty}{\|\lambda\|_1} < 1 - \exp[-1]$ and set constants ϵ, δ such that $\epsilon < 1 - \frac{\|\lambda\|_\infty}{\|\lambda\|_1}$, and $\delta < (\exp[-1] - \epsilon)\|\lambda\|_1$. Then, it holds that:*

$$\text{LiDAR}(\tilde{e}) \leq \text{LiDAR}(e) \exp\Big[-2p \log\Big(\frac{\|\lambda\|_1}{\|\lambda\|_1 + r\delta}\Big) - \frac{r\delta}{\|\lambda\|_1}\log(\frac{\delta}{\|\lambda\|_1})\Big] \tag{4}$$

Proof. From the independence of the noise vector μ, it is easy to see that:

$$\Sigma_b(\tilde{e}) = \begin{pmatrix} \Sigma_b(e) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \quad \Sigma_w(\tilde{e}) = \begin{pmatrix} \Sigma_w(e) & \mathbf{0} \\ \mathbf{0} & \Sigma_\mu \end{pmatrix}, \quad \Sigma_{\text{lidar}}(\tilde{e}) = \begin{pmatrix} \Sigma_{\text{lidar}}(e) & \mathbf{0} \\ \mathbf{0} & \delta I_r \end{pmatrix} \tag{5}$$

Then, we have that:

$$\textit{LiDAR}(\tilde{e}) = \exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big) + \mathcal{R}\Big] \tag{6}$$

where:

$$\mathcal{R} = \exp\Big[-r\Big(\frac{\delta}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\log\Big(\frac{\delta}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\Big] \tag{7}$$

$$\leq \exp\Big[-\frac{r\delta}{\|\lambda\|_1}\log(\frac{\delta}{\|\lambda\|_1})\Big] \tag{8}$$

where we used the fact that $\frac{\delta}{\|\lambda\|_1 + r\delta} + \epsilon < \exp[-1]$ by assumption to deduce the maximum of the function $|x|$ in the interval $x \in (0, \exp[-1])$. Note that:

$$\exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\Big] \tag{9}$$

$$= \exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + \delta r} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1} + \epsilon\Big) \tag{10}$$

$$-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + \delta r} + \epsilon\Big)\log\Big(\frac{\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon}{\frac{\lambda_i}{\|\lambda\|_1} + \epsilon}\Big)\Big] \tag{11}$$

Since $\forall_i,\ \frac{\lambda_i}{\|\lambda\|_1} + \epsilon < 1$ by assumption, we have:

$$\exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + \delta r} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1} + \epsilon\Big)\Big] \tag{12}$$

$$\leq \exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1} + \epsilon\Big)\Big] = \textit{LiDAR}(e) \tag{13}$$

hence we can write:

$$\exp\Big[-\sum_{i=1}^{p}\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\log\Big(\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon\Big)\Big] \tag{14}$$

$$\leq \textit{LiDAR}(e)\exp\Big[-p\Big(1 + \epsilon\Big)\min_i \log\Big(\frac{\frac{\lambda_i}{\|\lambda\|_1 + r\delta} + \epsilon}{\frac{\lambda_i}{\|\lambda\|_1} + \epsilon}\Big)\Big] \tag{15}$$

$$\leq \textit{LiDAR}(e)\exp\Big[-2p\log\Big(\frac{\|\lambda\|_1}{\|\lambda\|_1 + r\delta}\Big)\Big] \tag{16}$$

Combining equations 8 and 16, we get the result:

$$\textit{LiDAR}(\tilde{e}) \leq \textit{LiDAR}(e)\exp\Big[-2p\log\Big(\frac{\|\lambda\|_1}{\|\lambda\|_1 + r\delta}\Big) - \frac{r\delta}{\|\lambda\|_1}\log(\frac{\delta}{\|\lambda\|_1})\Big] \tag{17}$$

□

Note that an immediate consequence of Proposition 1 is that for $\delta << \|\lambda\|_1$ we have that *LiDAR*$(\tilde{e}) \leq$ *LiDAR*.

Eigenspectrum Decay As additional motivation, we invoke the arguments made in (Stringer et al., 2019) and later expanded in (Agrawal et al., 2022b) on the optimal eigenspectrum decay rate in an infinite dimensional embedding space, given by $\lambda_i \sim \Theta(i^{-1})$. In (Stringer et al., 2019), it was shown that a slower decay rate would necessarily imply a non-smooth kernel function, which, in turn, implies a non-monotonic relationship between rank and downstream performance. *LiDAR* circumvents this issue by implementing whitening operation through the inverse of Σ_w. This, in theory, permits the attainment of both a smooth embedding and the flexibility for eigenvalue decay in Σ_{LDA} to occur at a very gradual rate. It is essential to acknowledge, however, that while the smoothness can be maintained, an excessively high LDA rank may, in theory, have adverse consequences on downstream performance, owing to other underlying factors. We, therefore, defer a more comprehensive theoretical exploration of the implications of *LiDAR* to future research endeavors.

C Experimental Details

LiDAR is intended to serve as a proxy metric to compare the quality of representation as they relate to downstream performance. This task is intrinsically challenging due to the inherent uncertainty associated with the nature of the downstream task. In this paper, we focus on downstream classification tasks, employing the widely adopted linear probing protocol. A robust metric in this sense is one that consistently replicates the downstream performance ranking observed with a ground truth oracle across a set of pre-trained models. As in existing solutions, we only compare models from the same class, each one pre-trained using a different set of predefined hyperparameters. It is important to emphasize that a linear correlation between the metric and the oracle is not implied, hence we resort to established statistical tests which assess the strength and direction of the monotonic relationship between two variables. In other words, we seek to measure how well the relationship between two variables, the metric and the oracle, can be described using a monotonic function. We note that such evaluations go beyond simply picking the best model according the metric of interest out of a set of models, due to possible outliers in the set.

Spearman's rank correlation coefficient The Spearman correlation coefficient (Spearman, 1987, 1961) is essentially the Pearson correlation coefficient computed on the ranked values of two variables. While Pearson's correlation assesses linear relationships, Spearman's correlation evaluates monotonic relationships, which can be linear or non-linear in nature. In cases where there are no duplicate data values, a perfect Spearman correlation of either +1 or -1 indicates that each of the variables exhibits a flawless monotonic relationship with the other, forming a perfect monotone function. Given two sequences of real numbers $X = x_1, ..., x_n$ and $Y = y_1, ..., y_n$ and their corresponding ranking $R(X), R(Y)$, the Spearman correlation coefficient is given by:

$$r_s = 1 - \frac{d \sum_i (R(x_i) - R(y_i))^2}{n(n^2 - 1)} \tag{18}$$

Kendall's Tau rank correlation coefficient The Kendall's Tau correlation coefficient (Kendall, 1938b,a) is another popular alternative the to Spearman rank correlation, and is known to be superior when the sample size is small and has many tied ranks. The Kendall's Tau coefficient uses concordant and discordant pairs in its measure. For indices $j > i$, the pairs $\{x_i, x_j\} \in X, \{y_i, y_j\} \in Y$ are said to be concordant if the sort order of both pairs agree, otherwise they are said to be discordant. Let C and D denote the number of concordant and discordant pairs in X, Y then the Kendall's τ is given by $|C - D|/(C + D)$.

Top ranking model In similar spirit to (Garrido et al., 2022), as an additional evaluation we report the top ranking model given a metric of interest, and its downstream performance, and compare it to the optimal model according to the oracle. This protocol can be seen as a noisy estimate of Kendall's Tau rank correlation coefficient.

C.0.1 Models, Hyperparameters and Data

We use 5 different multiview JE SSL methods, spanning contrastive and regularized methods, as well as ResNet and transformer based. We use I-JEPA (Assran et al., 2023) and data2vec (Baevski et al., 2022) as representative of more recent masking based approaches that are not reliant on domain specific data augmentation. We use SimCLR (Chen et al., 2020) as a representative of contrastive methods, while we use DINO (Caron et al., 2021) as an example of self-distillation method and VICReg (Bardes et al., 2021) as representatives of regularized methods. Note that I-JEPA and data2vec use Transformer based encoders by design. We use a vision transformer (ViT) (Dosovitskiy et al., 2021) based encoder for DINO as well, and ResNet-50 (He et al., 2016) based encoders for SimCLR and VICReg. We vary different hyperparameters per method. The varied hyperparameters range from optimization related ones such as learning rate, and weight decay, architecture specific hyperparameters such as softmax temperature, and data augmentation and masking based hyperparameters. We stress that some SSL objectives such as I-JEPA and data2vec, which rely on specific forms of input masking, are extremely sensitive to the the masking hyperparameters, hence providing an important testbed for LiDAR. We highlight the drawback of conducting a grid search over a single hyperparameter, due to the fact that all the remaining frozen ones are typically highly optimized to the task, offering a somewhat contrived testbed. Hence, in addition to a standard grid search, we use random search over all hyperparameters. This is done by uniformly sampling each hyperparameter from a fixed range, providing a better cover for the space. We use the Imagenet-1k dataset (Russakovsky et al., 2015) for all experiments. We use the train split as the source dataset for pretraining and linear probing, and use the test split as the target dataset. For each pretrained checkpoint, we train a linear probe on the train split, which we denote as the oracle, and record its test performance on the test split.

C.0.2 Implementation Considerations

The implementation of LiDAR entails computing empirical approximations to Σ_w, Σ_b, which differs from one SSL method to another due to the inherent differences in the way input pairs are sampled. As a general rule, for each SSL method we use its own input transformations without alterations. In asymmetrical architectures, we compare both branches, denoted as the ”student” and the ”teacher” for evaluation. In transformer based architectures (as employed by design in I-JEPA, data2vec) we pool the final embedding/representation to produce vectors. For methods such as I-JEPA and data2vec, computing the *RankMe* score on the embeddings is not a straightforward task. This is due to the fact that the projector function’s task in both does not serve as a representation expander, rather it serves as a conditional predictor, predicting masked representations, conditioned on the spacial locations of the masked patches. For these methods, we evaluate *RankMe* on the student or teacher encoder instead. For SimCLR and VICReg, we copy the implementation details from (Garrido et al., 2022). All model are trained for a maximum of 300 epochs. Finally, in our analysis within the context of *RankMe*, we have noticed that utilizing the same data augmentations for both training and feature covariance matrix computation consistently leads to improved performance. In our experiments, we provide empirical results for both ”vanilla” and augmented *RankMe* as baselines.

C.1 Compute Considerations

While calculating the *LiDAR* score is a straightforward and computationally efficient procedure, it does introduce an additional computational load when compared to standard covariance estimation. This additional load arises from the need to perform matrix inversion in $\Sigma_w^{-0.5}$, which can be, theoretically, time-consuming when dealing with high-dimensional embeddings. In our experiments, we have observed that this computational overhead is generally inconsequential for all tested models. It tends to be overshadowed by the computational demands of the forward process required to generate the features. Nonetheless, we note the following trivial bound on the rank of Σ_{lidar}:

$$\text{Rank}(\Sigma_{\text{lidar}}) \leq \min\big(\text{Rank}(\Sigma_w), \text{Rank}(\Sigma_b)\big) \tag{19}$$

Since the rank of $\text{Rank}(\Sigma_b)$ is bounded by the number of surrogate classes used n (number of ”clean” samples used to sample), simple dimensionality reduction can be used when $p >> n$ to reduce the rank of Σ_w before inversion, without any loss in performance. We find in our experiments that $n = 1k, q = 50$ is sufficient to saturate performance, with the exception of VICreg, which required $n = 5k$ (and $q = 10$ to maintain a total of 50 features).

D Limitations

While we observe *LiDAR* significantly improves upon *RankMe* in most experiments, it is important to emphasize its drawbacks. Notably, we have observed instances where the LiDAR metric exhibits a negative correlation with probe accuracy, particularly pronounced in scenarios like VICReg when dealing with higher dimensional embeddings. This phenomenon underscores the intrinsic complexity of the relationship between rank, however it is measured, and downstream task performance. It serves as a reminder that these two factors are not necessarily causally linked; a high rank does not guarantee superior performance. Moreover, it's essential to acknowledge the computational overhead associated with calculating the LiDAR metric, which often involves the inversion of high-dimensional matrices. This computational complexity adds to the method's overall computational cost. Consequently, the feasibility of incorporating LiDAR as a loss signal for pretraining should be carefully considered, as it could be prohibitively expensive to naively evaluate at each iteration of the training process. Due to the sheer volume of experiments and necessary compute, we focus solely on linear probing in this work, using Imagenet-1K train-test split as source and target datasets. We expect *LiDAR*'s impressive performance to carry over to nonlinear probing protocols, as well as OOD domains, however we relegate such investigations to future work. Lastly, it's worth noting that our approach is contingent on the sampling strategy for positive pairs x and $\tilde{x}$ which varies among methods, making cross-method comparisons challenging.

E Implementation Details

E.1 VICReg

VICReg (Bardes et al., 2021) proposes an explicit regularization to prevent dimensional collapse in self-supervised learning methods. VICReg (Bardes et al., 2021) consists of the standard invariance term, a variance term to encourage networks to not suffer from dimensional collapse and a covriance term that encourages the covariance matrix of embeddings to be approximately diagonal. Let $z_i \in \mathbb{R}^d$ denote the features from a neural network. The variance term is given by:

$$var(Z) = \frac{1}{d} \sum_{j=0}^{d-1} \max\left(0, 1 - S\left(z_j, \epsilon\right)\right) \tag{20}$$

where $S\left(x, \epsilon\right)$ denotes the standard deviation and ϵ is a small value to prevent numerical instability. The covariance term is given by:

$$c(Z) = \frac{1}{d} \sum_{i \neq j} \left[C(Z]_{i,j}^2 \tag{21}$$

while the invariance term is the L_2 distance between the usual positive data pairs used to optimize the network. The complete loss function used to optimize a network in VICReg is given by:

$$L\left(Z, Z'\right) = \frac{\lambda}{n} \sum_i \|z_i - z_j\| + \mu \left[var\left(Z\right) + var\left(Z'\right)\right] + \nu \left[c\left(Z\right) + c\left(Z'\right)\right] \tag{22}$$

where λ, μ and ν are hyperparameters that control the contribution from the invariance, standard deviation and covariance terms respectively.

We use the reference implementation [6] provided by Bardes et al. (2021) to train a ResNet-50 (He et al., 2016) backbone on Imagenet-1K dataset (Russakovsky et al., 2015). The projector used is a standard multi-layer perceptron (MLP) with dimensions 8192-8192-8192. We train a ResNet-50 with VICReg for 100 epochs using hyperparameters and training protocol described in (Bardes et al., 2021). The trained backbone is probed by trianing a classifier on the frozen features of the backbone. We use a setup that is described in *RankMe* including data preprocessing and optimizer-related hyperparameters to train a probe for 30 epochs. We use 32 hyperparametr sets that include described in Garrido et al. (2022) in our experiments.

We use ImageNet-1K (Russakovsky et al., 2015) training data as our source dataset for self-supervised learning (SSL). We use 10000 images from the source dataset, i.e., ImageNet-1K training split and 10 augmentations per image to construct a labeled dataset needed to calculate *LiDAR* and augmented-*Rankme* . We use 25600 images from the source dataset to calculate *RankMe*. All pipelines that calculate various metrics employ the exact data augmentation pipeline used for SSL pretraining. The results for VICReg are as follows:

- We calculate the *LiDAR*, *RankMe*, augmented-*RankMe* for the final checkpoint after SSL training. Figure 3 shows a scatter plot of the 32 checkpoints where we plot the various effective rank metrics versus probe accuracy.
- We calculate the effective rank metrics on checkpoints collected every 20 epochs during training. Figure 4 shows the evolution of the metrics during training.
- Table 4 shows the correlations estimated via Spearman rank correlation and Kendall's τ correlation tests for all checkpoints collected during training.

[6] `https://github.com/facebookresearch/vicreg`

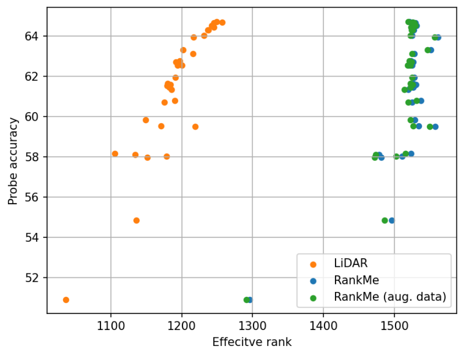


Figure 3: VICReg: Performance of VICReg representations measured by RankMe and LiDAR. Each point refers to a checkpoint evaluated after 100 epochs of self-supervised pretraining. LiDAR shows strong correlation

Correlation	RankMe	RankMe (aug. dataset)	LiDAR
Spearman rank	0.3174	0.5209	**0.9161**
Kendall's τ	0.2056	0.3790	**0.8105**

Table 4: VICReg: Compare RankMe and LiDAR using Spearman Rank correlation and Kendall's τ correlation measures after 100 epochs of training. VICReg representations are used to estimate *RankMe* and *LiDAR* for the 32 hyperparameter sets considered in our experiments. The hyperparamter values are identical to the values considered by (Garrido et al., 2022)

Metric	cov.	inv.	LR	WD
Imagenet Oracle	64.7380	62.7800	63.9500	61.6500
RankMe	64.5400	59.5400	**63.9500**	**59.5200**
RankMe (aug. dataset)	64.5400	59.5400	**63.9500**	**59.5200**
LiDAR	**64.7080**	**62.5720**	**63.9500**	**59.5200**

Table 5: VICReg: Linear probe accuracy recovered by *RankMe*, augmented-*RankMe* and *LiDAR* on ImageNet-1K dataset at the end of training. The metrics presented above are calculated with representations

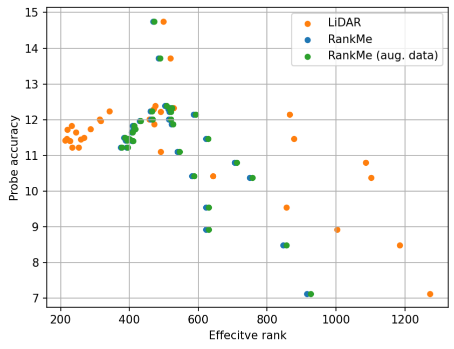
Epoch	RankMe	RankMe (aug. dataset)	LiDAR
20	0.6081	0.6466	**0.6957**
40	0.4315	0.5949	**0.8699**
60	0.4065	0.5381	**0.8809**
80	0.3904	0.5905	**0.9032**
100	0.3174	0.5209	**0.9161**

(a) VICReg: Spearman Rank coefficient

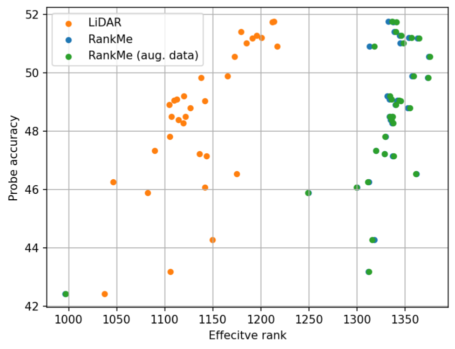
Epoch	RankMe	RankMe (aug. dataset)	LiDAR
20	0.4476	0.4718	**0.5323**
40	0.2984	0.4597	**0.7097**
60	0.2702	0.3750	**0.7218**
80	0.2823	0.4435	**0.7823**
100	0.2056	0.3790	**0.8105**

(b) VICReg: Kendall's τ coefficient

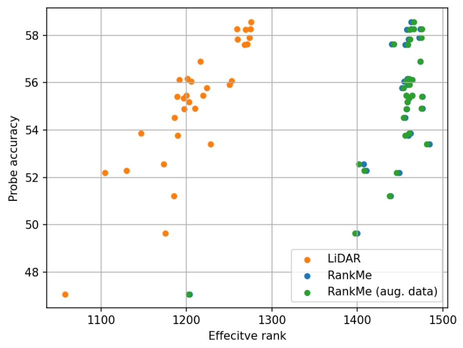
Table 6: VICReg: Correlation between effective rank estiamted by *RankMe* and *LiDAR* and probe accuracy evolution during training. Each row corresponds to a checkpoint collected at epoch specified in the table.



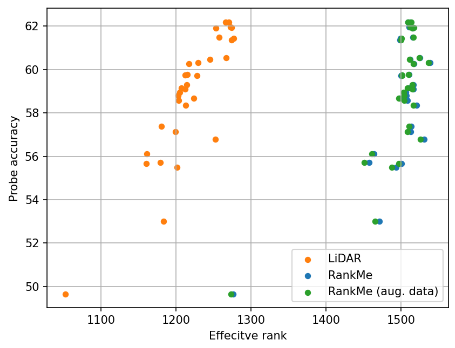
(a) 1 training epoch



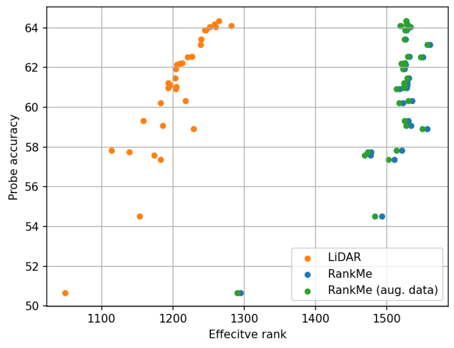
(b) 20 training epochs



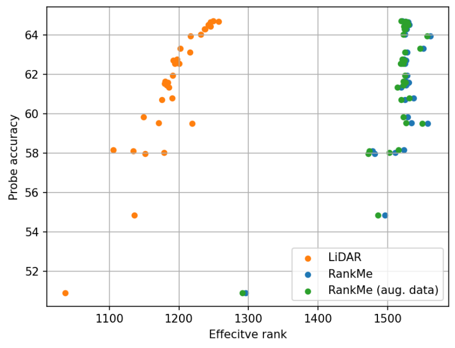
(c) 40 training epochs



(d) 60 training epochs



(e) 80 training epochs



(f) 100 training epochs

Figure 4: VICReg: Performance of VICReg representations measured by RankMe and LiDAR. Each point represents a row of hyperparameters among the 32 sets of hyperparameters consdiered in our experiments. The hyperparamter values are identical to the values considered by (Garrido et al., 2022)

Epoch	RankMe	RankMe (aug. dataset)	LiDAR
20	0.6081	0.6466	**0.6957**
40	0.4315	0.5949	**0.8699**
60	0.4065	0.5381	**0.8809**
80	0.3904	0.5905	**0.9032**
100	0.3174	0.5209	**0.9161**

(a) VICReg: Spearman Rank coefficient

Epoch	RankMe	RankMe (aug. dataset)	LiDAR
20	0.4476	0.4718	**0.5323**
40	0.2984	0.4597	**0.7097**
60	0.2702	0.3750	**0.7218**
80	0.2823	0.4435	**0.7823**
100	0.2056	0.3790	**0.8105**

(b) VICReg: Kendall's τ coefficient

Table 7: VICReg: Correlation between effective rank estimated by *RankMe* and *LiDAR* and probe accuracy evolution during training. Each row corresponds to a checkpoint collected at epoch specified in the table.

Metric	cov.	inv.	LR	WD
Imagenet Oracle	64.7380	62.7800	63.9500	61.6500
RankMe	64.5400	59.5400	**63.9500**	**59.5200**
RankMe (aug. dataset)	64.5400	59.5400	**63.9500**	**59.5200**
LiDAR	**64.7080**	**62.5720**	**63.9500**	**59.5200**

Table 8: VICReg: (b) Linear probe accuracy recovered by *RankMe*, augmented-*RankMe* and *LiDAR* on ImageNet-1K dataset at the end of training. The metrics presented above are calculated with representations.

E.2 I-JEPA

Image-based Joint-Embedding Predictive Architecture (I-JEPA) (Assran et al., 2023) is a recently proposed non-generative approach to learn semantically strong representations in a self-supervised manner. I-JEPA splits an image into a context block and several target blocks and uses the context block to predict the target blocks. Note that each block is composed of image patches. The core innovation in I-JEPA is the design of a masking strategy that is shown to lead to semantically strong representations when used in conjunction with Vision Transformers (ViTs) (Dosovitskiy et al., 2021). I-JEPA uses a ViT to encode the non-masked context patches and another ViT to predict the encodings for the masked out target patches. The target representations are provided by a target encoder which is an exponential moving average (EMA) version of the context encoder. In this work we refer to the context encoder as the student encoder and the target encoder as the teacher and use these terms interchangeably in our presentation. The loss function is applied to the embeddings that are output by the student and the teacher and is given by:

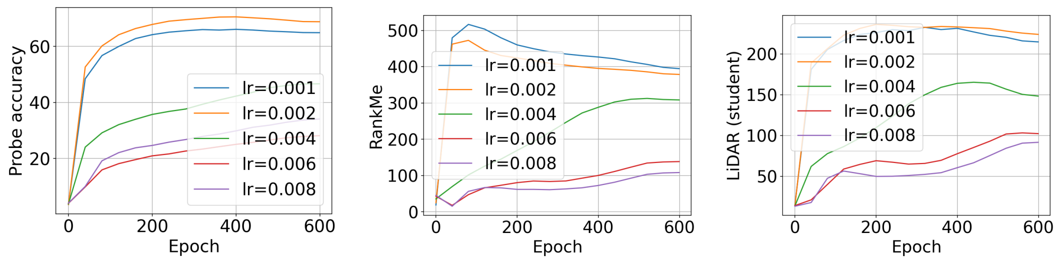
$$\frac{1}{M}\sum_{n=1}^{M}\sum_{i\in B_i}\|y_i-\hat{y}_i\|^2 \tag{23}$$

where M denotes the number of target blocks, B_i denotes a set of block indices in a target and y and $\hat{y}$ denote the embeddings provided by the student and the teacher respectively. The asymmetry introduced due to student and teacher encoders allows I-JEPA to avoid representation collapse (Assran et al., 2023).

In order to apply ***LiDAR*** for I-JEPA Assran et al. (2023) we create a labeled dataset by first selecting a fixed number of images at random and treat each image as a class. We then apply the masking approach proposed in I-JEPA (Assran et al., 2023) to create multiple instances of a class to create a labeled dataset needed to calcualte the linear discriminant analysis matrix for *LiDAR*. The embeddings y and $\hat{y}$ described in 23 are used as inputs to estimate Student and Teacher *LiDAR* measures. We use the output of the student encoder to calculate *RankMe* measure. As described in Section C.0.2 the embeddings y and $|haty$ are used to calculate the augmented *RankMe* metric which is denoted as *RankMe* (aug.) in the results. We use 1000 images and 50 augmentations to construct the labeled dataset for *LiDAR* and augmented-*RankMe* while we use 10000 image samples for *RankMe*. Self-supervised training is run for 600 epochs with an effective batch size of 2048 using the training protocol described in I-JEPA (Assran et al., 2023). The downstream task consists of linear probing frozen representations on the ImageNet-1K dataset (Russakovsky et al., 2015). The probe is optimized with Adam (Kingma & Ba, 2015) optimizer for 20 epochs with a starting learning rate of 0.01 and a step learning rate schedule where the base learning rate is dropped by a factor 10 after 15 epochs. The following hyperparamter sets are used in our experiments to empirically estimate the performance of *RankMe* augmented-*RankMe* and *LiDAR*:

- Learning rate from $0.001, 0.002, 0.004, 0.006$ and 0.008. Figure 5, Figure 6 show the results of experiments where we vary the learning rate parameter alone. Table 11 and Table 1 show the rank correlation coefficients while Table 2 show the accuracy recovered by *RankMe* and *LiDAR*.
- Weight decay from $0.05, 0.1, 0.2$and 0.4. Figure 7, Figure 8 show the results of experiments where we vary the weight decay parameter alone. Note that the weight decay is kept fixed throughought training in this set of experiments. Table 11 and Table 1 show the rank correlation coefficients while Table 2 show the accuracy recovered by *RankMe* and *LiDAR*.
- Target mask scale factor from $[0.15, 0.2]$, $[0.125, 0.2]$, $[0.2, 0.25]$ and $[0.2, 0.3]$. Figure 9, Figure 10 show the results of experiments where we target mask scale ratio alone. Table 11 and Table 1 show the rank correlation coefficients while Table 2 show the accuracy recovered by *RankMe* and *LiDAR*.
- Target mask scale factor from $[0.85, 1.0]$, $[0.75, 1.0]$, $[0.65, 1.0]$ and $[0.4, 1.0]$. Figure 11, Figure 12 show the results of experiments where we target mask scale ratio alone. Table 11 and Table 1 show the rank correlation coefficients while Table 2 show the accuracy recovered by *RankMe* and *LiDAR*.

Additionally, we conduct a random hyperparameter search experiment where we sample 20 sets of hyperparamters uniformly and train a ViT-B (Dosovitskiy et al., 2021) with I-JEPA (Assran et al., 2023). The hyperparametrs were sampled from the following uniform distributions:



(a) Linear probe accuracy (b) *RankMe* (c) *LiDAR (student)*

Figure 5: I-JEPA: ViT-Base architecture trained on Imagenet-1K by varying the learning rate. Plots show the evolution of metrics over training time.

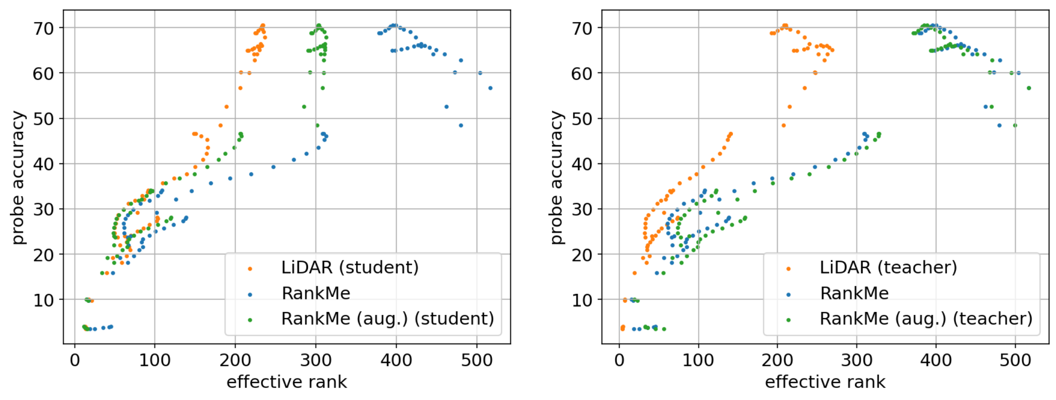
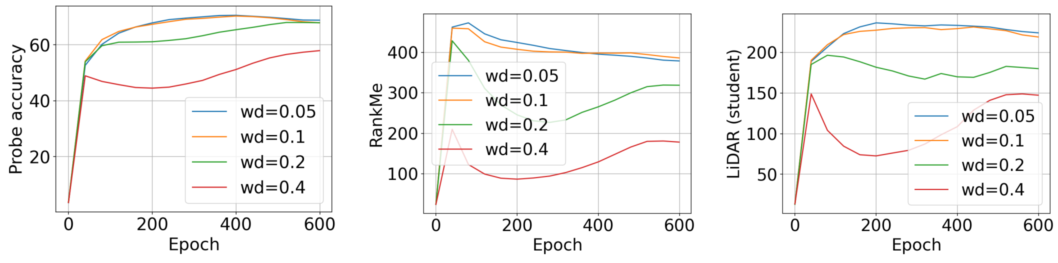


Figure 6: I-JEPA: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K by varying the learning rate.

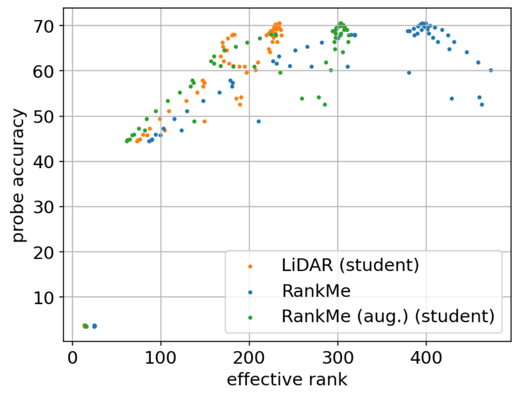
- learning rate from $[0.0001250.0002]$
- weight decay from $[0.050.4]$
- target scale (minimum) from $[0.10.2]$
- target scale (maximum) from $[0.20.4]$
- context scale (minimum) from $[0.30.95]$

The results of these experiments are shown in Figure 2 and the correlations are available in Table 1 and Table 12. The probe accuracy recovered for hyperparameters generated via random search are shown in Table 2.



(a) Linear probe accuracy (b) *RankMe* (c) *LiDAR (student)*

Figure 7: I-JEPA: ViT-Base architecture trained on Imagenet-1K by varying the weight decay. Plots show the evolution of metrics over training time.



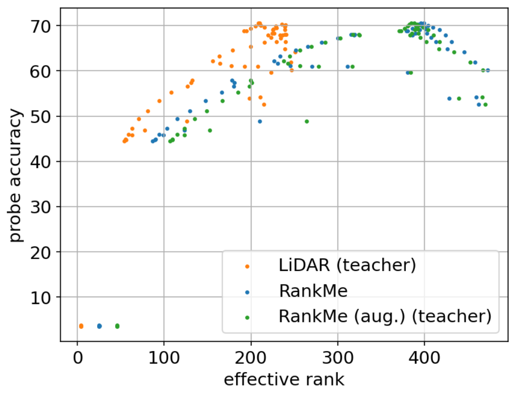


Figure 8: I-JEPA: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K by varying weight decay.

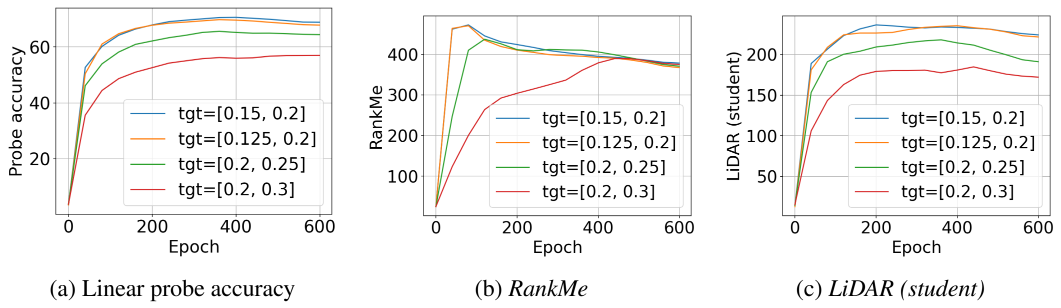
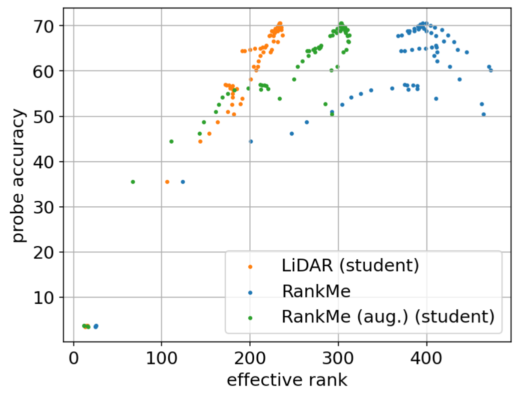


Figure 9: I-JEPA: ViT-Base architecture trained on Imagenet-1K by varying the target mask scale hyperparameter. Plots show the evolution of metrics over training time.



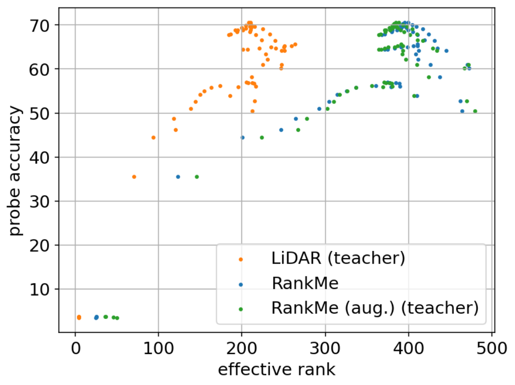
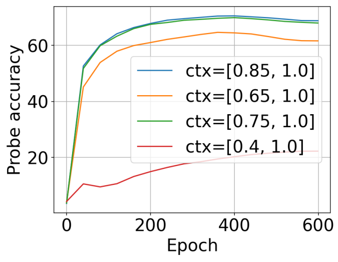
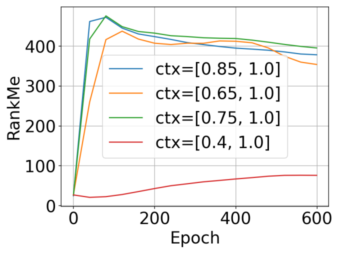


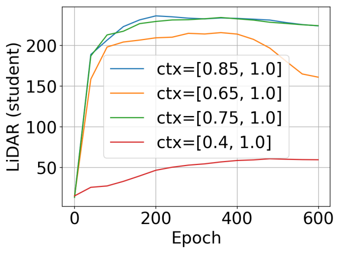
Figure 10: I-JEPA: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K by varying the target mask scale factor.



(a) Linear probe accuracy



(b) *RankMe*



(c) *LiDAR (student)*

Figure 11: I-JEPA: ViT-Base architecture trained on Imagenet-1K by varying the context mask scale hyperparameter. Plots show the evolution of metrics over training time.

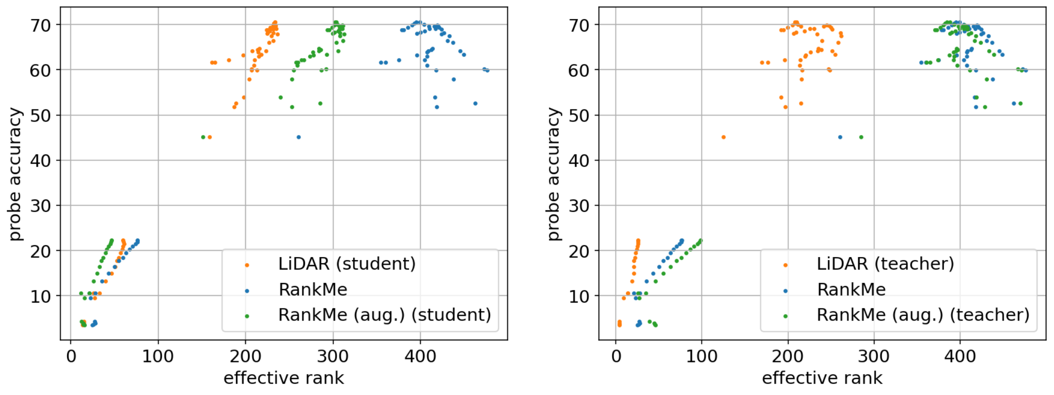


Figure 12: I-JEPA: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K by varying the context mask scale factor.

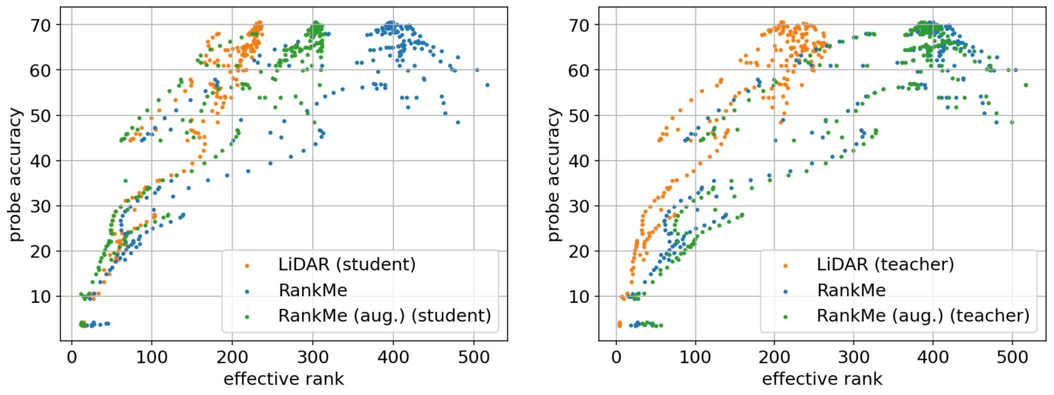


Figure 13: I-JEPA: Aggregated scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K. Hyperparameters varied are one of learning rate, weight decay, target mask scale or context mask scale. Plots show that effective rank estimated by *LiDAR* has high correlation with linear probe accuracy.

Hyperparameter	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Learning rate	0.6835	0.7829	**0.8443**	0.6734	0.7506
Weight decay	0.5040	0.7034	**0.7331**	0.4792	0.5278
Target mask scale	0.2867	0.6786	**0.7937**	0.2956	0.2927
Context mask scale	0.4246	0.7867	**0.8482**	0.3929	0.5556
Overall	0.5830	0.7513	0.8159	0.5713	0.6828

Table 9: I-JEPA: Kendall's τ coefficient between effective ranks of RankMe, RankMe (aug.) and LiDAR and linear probe accuracy. Hyperparameters are varied via grid search.

Metric	LR	WD	Target mask scale	Context mask scale	Overall
ImageNet Oracle	70.5800	70.5800	70.5800	70.5800	70.5800
RankMe	56.7580	60.2040	60.2040	59.8960	59.8960
RankMe (aug.) (Student)	**67.8680**	**67.8680**	**67.8680**	**67.8680**	**67.8680**
RankMe (aug.) (Teacher)	56.7580	52.6720	50.4200	59.8960	59.8960
LiDAR (Student)	**67.8680**	**67.8680**	**67.8680**	**67.8680**	**67.8680**
LiDAR (Teacher)	65.1080	64.1920	65.5820	67.5420	67.5420

Table 10: I-JEPA: Linear probe accuracy recovered by *RankMe* and *LiDAR* on ImageNet-1K dataset. Hyperparameters set via grid search.

Hyperparameter	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Learning rate	0.8775	0.9258	**0.9605**	0.8747	0.9030
Weight decay	0.6381	0.8669	**0.8884**	0.6196	0.6978
Target mask scale	0.4008	0.8429	**0.9407**	0.4069	0.3851
Context mask scale	0.5800	0.9180	**0.9590**	0.5533	0.6940
Overall	0.7700	0.9104	**0.9494**	0.7627	0.8475

Table 11: I-JEPA: Spearman rank correlation coefficient for I-JEPA between effective ranks of RankMe, RankMe (aug.) and LiDAR and linear probe accuracy.

	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Random search	0.9470	**0.9770**	0.9511	0.9486	0.9722

Table 12: I-JEPA: Spearman rank correlation coefficient between effective ranks of *RankMe*, augmented-*RankMe* and *LiDAR* and linear probe accuracy. Hyperparmaeters are generated via random sampling.

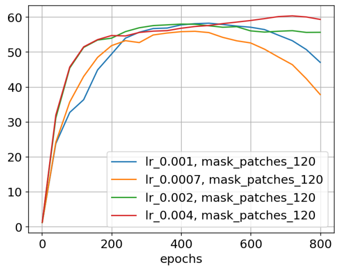
Table 13: data2vec: (a) Kendall's τ correlation coefficient and (b) linear probe accuracy. Table 15 and Table 16 shows a detailed breakdown

(a)

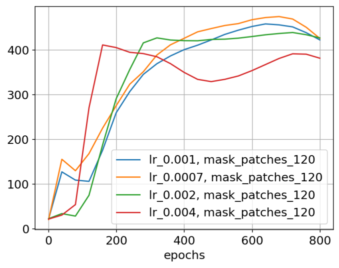
RankMe	RankMe (aug.) (Student, Teacher)	LiDAR (Student, Teacher)
0.2238	(0.2799, 0.2656)	(**0.5531**, 0.5227)

(b)

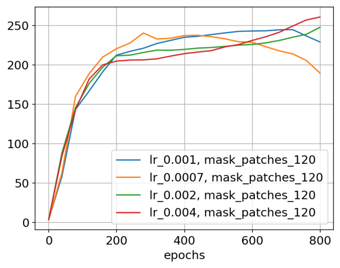
ImageNet Oracle	RankMe	RankMe (aug.) (Student, Teacher)	LiDAR (Student, Teacher)
60.39	48.60	(51.45 51.45)	(**59.37**, **59.37**)



(a) Linear probe accuracy



(b) *RankMe*



(c) *LiDAR (student)*

Figure 14: data2vec: ViT-Base architecture trained on Imagenet-1K by varying the learning rate. Plots show the evolution of metrics over training time.

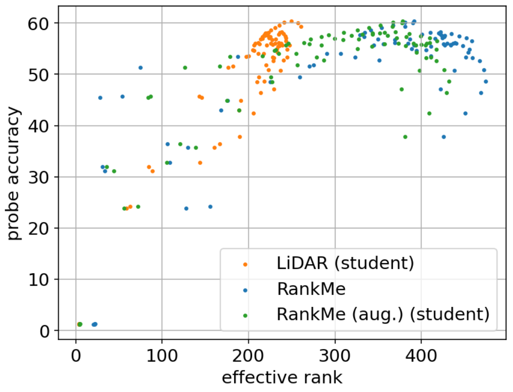
E.3 data2vec

data2vec (Baevski et al., 2022) is a self-supervised learning approach that aims to predict latent representations of input data based on a masked view of the input data. The idea behind data2vec (Baevski et al., 2022) is similar to I-JEPA (Assran et al., 2023) but the details of masking and the predictor used to predict embeddings are different.

We follow the same protocol described for I-JEPA above in terms of constructing a labeled dataset for *LiDAR* and augmented-*RankMe* with 10000 samples from ImageNet-1K and apply 50 augmentations. The augmentations applied in this case are identical to the augmentation described in (Baevski et al., 2022) and available via a reference implementation provided by the authors [7]. We use 10000 source images to calculate *RankMe*. We train a ViT-B (Dosovitskiy et al., 2021) model for 800 epochs with an effective batch size of 2048 and with other hyperparameters described in (Baevski et al., 2022). During training, we save a checkpoint every 40 epochs for analysis and downstream linear probing. Linear probing is done on frozen representations of the encoder using LARS optimizer (You et al., 2017) with a starting learning rate of 0.01 and a step learning rate schedule that drops every 15 epochs by a factor of 10 with an effective batch size of 16384 samples. We consider the following SSL training hyperparameters in our experiments to test the performance of *LiDAR* and *RankMe* and its augmented variant:

- Learning rate from 0.0007, 0.001, 0.002 and 0.004. Figure 14 and Table 14 and Table 15 show the results from these experiments
- Set number of mask patches to either 120 (default) or 80 while we fix the learning rate to 0.0007. Figure 16, Table 14 and Table 15 show the results from these experiments
- Table 16 shows the probe accuracy recovered by the various effective rank metrics.

[7] `https://github.com/facebookresearch/data2vec_vision/tree/main/beit`



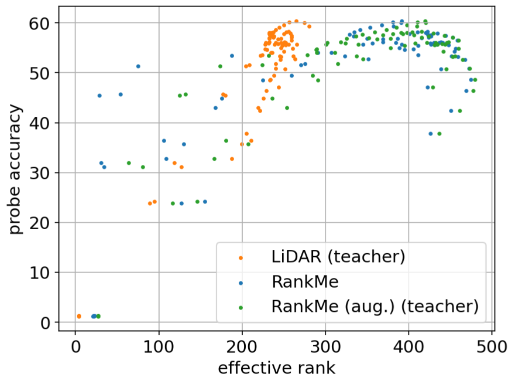
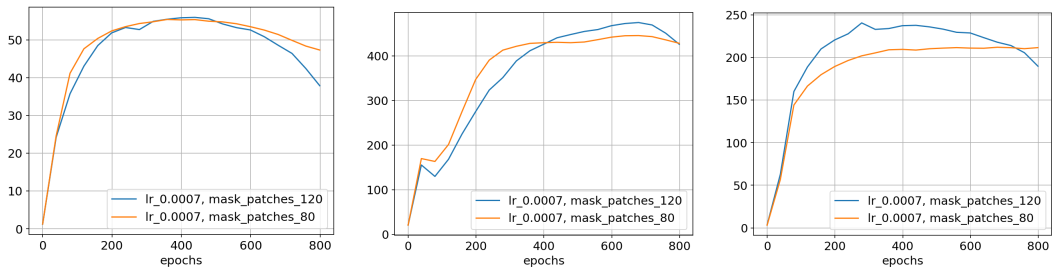
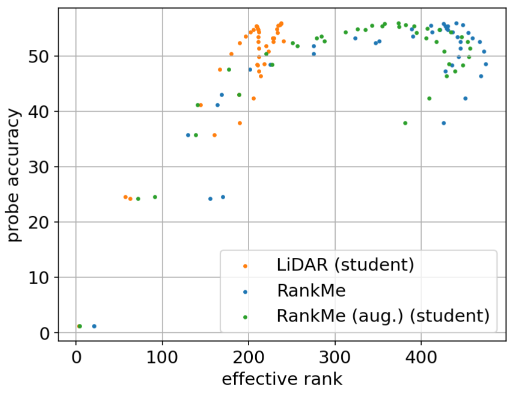


Figure 15: data2vec: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K. The learning rate was varied in this experiment.



(a) Linear probe accuracy (b) *RankMe* (c) *LiDAR (student)*

Figure 16: data2vec: ViT-Base architecture trained on Imagenet-1K by varying the masking ratio. Plots show the evolution of metrics over training time.



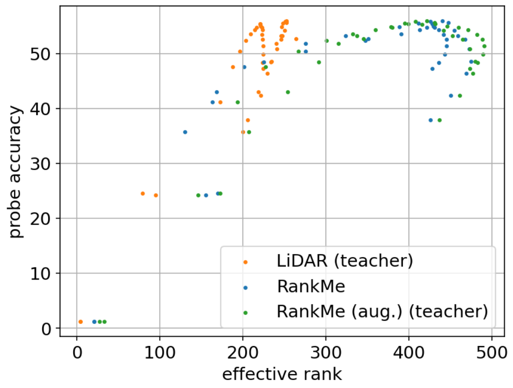
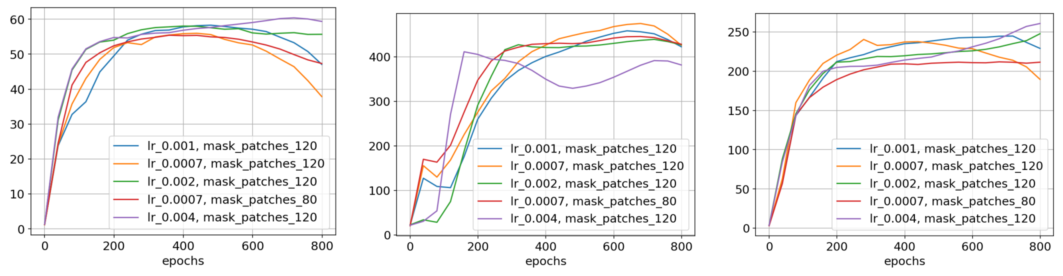


Figure 17: data2vec: Scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K. The masking ratio was varied in this experiment.



(a) Linear probe accuracy (b) *RankMe* (c) *LiDAR (student)*

Figure 18: data2vec: ViT-Base architecture trained on Imagenet-1K by varying one of learning rate or masking ratio. Plots show the evolution of metrics over training time.

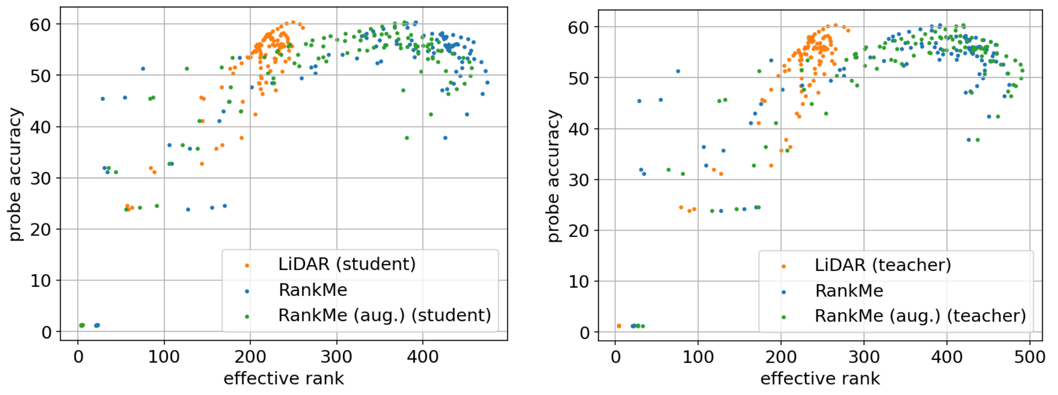


Figure 19: data2vec: Aggregated scatter plot of checkpoints collected during training of ViT-Base architecture on Imagenet-1K. Hyperparameters varied are one of learning rate and mask ratio (number of masked patches).

Hyperparameter	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Learning rate	0.3639	0.4440	**0.6680**	0.4191	0.6240
Mask ratio	0.3720	0.3026	**0.6089**	0.2785	0.5466
Overall	0.3408	0.3795	**0.7275**	0.3627	0.6962

Table 14: data2vec: Spearman rank correlation coefficient for data2vec between effective ranks of RankMe, RankMe (aug.) and LiDAR and linear probe accuracy.

Hyperparameter	RankMe	RankMe (aug.) (Student)	LiDAR (Student)	RankMe (aug.) (Teacher)	LiDAR (Teacher)
Learning rate	0.2410	0.3419	**0.5077**	0.3172	0.4716
Mask ratio	0.2683	0.2381	**0.4657**	0.2195	0.4170
Overall	0.2238	0.2799	**0.5531**	0.2626	0.5227

Table 15: data2vec: Kendall's τ for data2vec between effective ranks of RankMe, RankMe (aug.) and LiDAR and linear probe accuracy.

Metric	LR	Mask ratio	Overall
ImageNet Oracle	60.3920	55.9780	60.3920
RankMe	48.6040	48.6980	48.6980
RankMe (aug.) (Student)	48.6040	51.4500	51.4500
LiDAR (Student)	**59.3720**	**52.7460**	**59.3720**
RankMe (aug.) (Teacher)	48.6040	51.4500	51.4500
LiDAR (Teacher)	**59.3720**	**52.7460**	**59.3720**

Table 16: data2vec: Linear probe accuracy recovered by *RankMe* and *LiDAR* on ImageNet-1K dataset.

E.4 DINO

DINO (Caron et al., 2021) is an example of a self-distillation approach to learning representations in a self-supervised manner. DINO uses a student and teacher encoder (weights updated via exponential moving average) ,multiple-crops and small patches to train a ViT (Dosovitskiy et al., 2021) in a self-supervised manner. Evaluations on multiple tasks presented in DINO (Caron et al., 2021) show that the resulting encoder learns strong representations. The loss function for DINO (Caron et al., 2021) minimizes the cross-entropy between the probability distributions provided by the teacher and student network. We reproduce a description below from DINO (Caron et al., 2021) for completeness:

$$\min_{i} H\left(P_t\left(z\right), P_s\left(z\right)\right) \tag{24}$$

where H denotes the cross-entropy function, P_s and P_t denote the probability distributions output by the student and teacher networks respectively. The probabilities are computed via:

$$P_{net}(z)^{(i)} = \frac{\exp(f_{\theta_{net}}(z)^{(i)}/\tau_{net})}{\sum_{k=1}^{K}\exp(f_{\theta_{net}}(x)^{(k)}/\tau_{net})} \tag{25}$$

where the subscript "net" denotes either the student or the teacher network, f is a neural network parameterized by θ and $z^{(i)}$ is a feature vector for sample i. A key hyperparameter in DINO (Caron et al., 2021) is the temperature value τ_s and τ_t used to control the sharpness of the output distribution produced by the softmax function. We test the performance of *RankMe*, augmented-*RankMe* and *LiDAR* to predict linear probing performance by varying these parameters.

We train a ViT-S with a patch size of 16×16 using the protocol described by DINO (Caron et al., 2021) and implemented in a reference implementation provided by the authors of DINO [8] on the Imagenet-1K (Russakovsky et al., 2015) dataset. The projection head consists of a 3-layer MLP with hidden dimension of 2048 and an output dimension referred to as bottleneck dimension of 256 that provides embeddings. These embeddings are that projected to a higher dimensional space of 65536 in our experiments. We use the embeddings to estimate *RankMe*, augmented-*RankMe* and *LiDAR* in our experiments consistent with the methodology adopted in *RankMe* (Garrido et al., 2022). We use an effective batch size of 512 images and train the model for 300 epochs. Our experiments consists of:

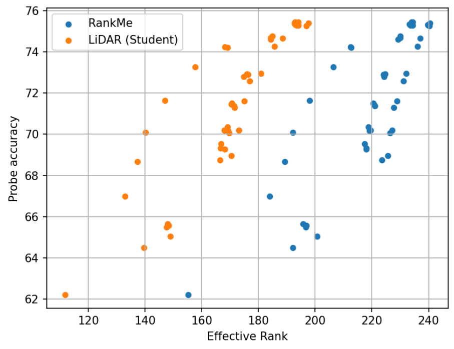
- varying the teacher temperature (τ_t) while keeping the student temperature (τ_s) fixed to 0.1. The values considered for (τ_t) are $\{0.02, 0.04, 0.06, 0.07\}$
- varying the student temperature (τ_s) while keeping the teacher temperature (τ_t) fixed. The values considered for (τ_s) are $\{0.2, 0.4\}$

We keep the rest of the training and linear probing hyperparameters identical to those provided in the official implementation [8]. The results of these experiments are available in Figure 20 and the correlation values are quantified in Table 17.

Correlation	RankMe	LiDAR (Student)	LiDAR (Teacher)
Spearman rank	0.8191	0.8807	0.8732
Kendall's τ	0.6288	0.7157	0.7299

Table 17: DINO: Compare *RankMe* and *LiDAR* with Spearman rank and Kendall's τ correlation coefficient metrics. Observe that *LiDAR* performs better than *RankMe* for both measures.

[8] `https://github.com/facebookresearch/dino`



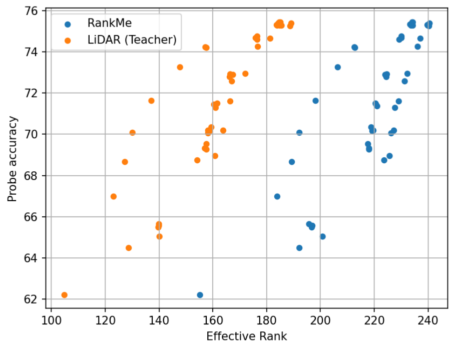


Figure 20: DINO: Aggregated scatter plot of checkpoints collected during training of ViT-Small architecture on Imagenet-1K. Hyperparameters varied are one of teacher or student softmax temperature described in the implementation section.

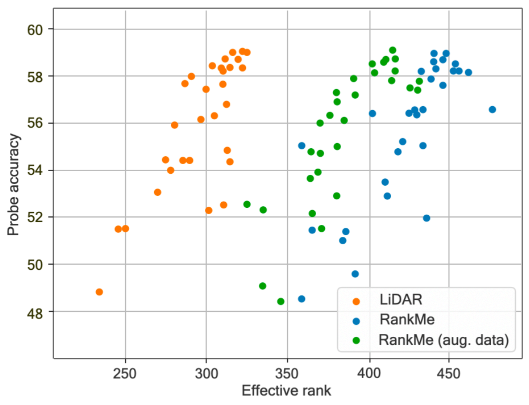


Figure 21: SimCLR: Performance of SimCLR representations measured by RankMe and LiDAR. Each point refers to a checkpoint evaluated with a **randomly searched** hyperparameter after 100 epochs of self-supervised pretraining. LiDAR shows strong correlation

E.5 SimCLR

SimCLR (Chen et al., 2020) is an example of a contrastive joint-embedding self-supservised learning method. The loss function for SimCLR is given by (Chen et al., 2020; Garrido et al., 2022) and included below for completeness:

$$L = -\sum_{i,j \in P} \frac{e^{Similarity(z_i, z_j)}}{\sum_{k=1}^{N} \mathbb{I}_{k \neq i} e^{Similarity(z_i, z_k)}} \tag{26}$$

where $Similarity$ denotes the cosine similarity between two vectors z_i and z_j, $\mathbb{I}$ denotes the indicator function, P is the set of all positive pairs and the number of examples in given by N.

We train a ResNet-50 backbone (He et al., 2016) with a 3 layer MLP projector with hidden dimensions 8192 and output dimension equal to 2048. In other words, the embeddings produced by SimCLR have a dimension equal to 2048. The network described above is trained with the LARS optimizer (You et al., 2017) and other settings described in (Chen et al., 2020) on the ImageNet-1k (Russakovsky et al., 2015) training split. The representations from the backbone are evaluated via standard linear probing by training a linear layer on ImageNet-1k training split and calculating test accuracy on the validation split. Probing is performed using SGD optimizer with Nesterov momentum with hyperparameters described in *RankMe* (Garrido et al., 2022). We consider two sets of experiments to test the performance of *LiDAR* and *RankMe* with SimCLR (Chen et al., 2020):

- A grid search with the hyperparametrs sets that include learning rate, weight decay, embedding dimension and softmax temperature used in *RankMe* (Garrido et al., 2022). Figure 22 shows a scatter plot of the probe accuracy vs. effective rank estiamted by *RankMe*, augmented-*RankMe* and *LiDAR*. Table 19 quantifies the correlations for the above methods. Table 21 shows the linear probe accuracy recovered by the various metrics and compares the results to ImageNet Oracle
- A random hyperparameter search by randomly sampling learning rate, weight decay and softmax temperature to generate hyperparameter sets. We create a table of hyperpameters with learning rate chosen from $[0.3, 0.4, 0.5, 0.6]$, weight decay chosen from $[10^{-7}, 10^{-6}, 10^{-5}]$ and softmax temperature chosen from $[0.05, 0.1, 0.15, 0.2, 0.25]$. We select 30 sets of hyperparameters from this table and use these to train models with SimCLR and evaluate performance. Figure 21 shows the results for this experiment as a scatter plot and Table 19 quantifies the correlations.

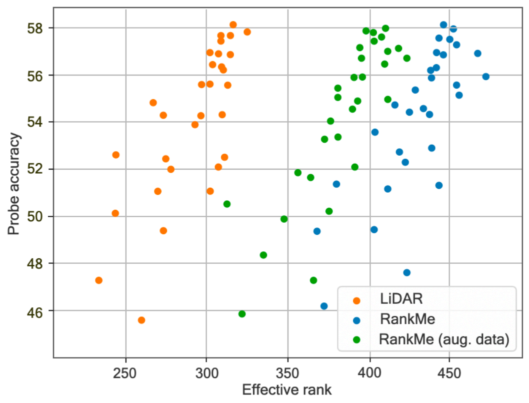


Figure 22: SimCLR: Performance of SimCLR representations measured by RankMe and LiDAR. Each point refers to a checkpoint evaluated with the **grid search** hyperparameter from Rankme paper after 100 epochs of self-supervised pretraining. LiDAR shows strong correlation

Hyperparameter	RankMe	RankMe (aug. dataset)	LiDAR
Learning rate	0.5248	0.5301	**0.9138**
Weight decay	0.4019	0.5573	**0.8964**
Temperature	0.5704	0.6764	**0.9182**
Overall	0.5125	0.6304	**0.9155**

(a) Spearman Rank correlation coefficient

Hyperparameter	RankMe	RankMe (aug. dataset)	LiDAR
Learning rate	0.5706	0.4694	**0.7512**
Weight decay	0.3580	0.4221	**0.7148**
Temperature	0.4209	0.7061	**0.8435**
Overall	0.4906	0.5581	**0.7967**

(b) Kendall's τ correlation coefficient

Table 18: SimCLR: Correlation between effective rank estiamted by *RankMe* and *LiDAR* and probe accuracy per hyperparameter.

Correlation	RankMe	RankMe (aug. dataset)	LiDAR
Spearman rank	0.5125	0.6304	**0.9155**
Kendall's τ	0.4906	0.5581	**0.7967**

Table 19: SimCLR: Compare RankMe and LiDAR using Spearman Rank correlation and Kendall's τ correlation measures evaluated during hyperparameter **grid search**.

Correlation	RankMe	RankMe (aug. dataset)	LiDAR
Spearman rank	0.5301	0.6389	**0.9188**
Kendall's τ	0.4982	0.5761	**0.8167**

Table 20: SimCLR: Compare RankMe and LiDAR using Spearman Rank correlation and Kendall's τ correlation measures evaluated during hyperparameter **random search**.

Metric	LR	WD	Temp.	Overall
Imagenet Oracle	58.2370	56.6740	57.4710	59.1420
RankMe	55.8950	55.1490	56.0390	56.4630
RankMe (aug. dataset)	57.2010	55.8170	56.3170	57.8260
LiDAR	**57.8940**	**56.3020**	**57.0920**	**58.9270**

Table 21: SimCLR: Linear probe accuracy recovered by *RankMe*, augmented-*RankMe* and *LiDAR* on ImageNet-1K dataset at the end of training. Results are from trials run with hyperparameter **random search**.