CONTRASTIVE LATENT REPRESENTATION LEARNING WITH DENOISING SCORE MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Contrastive representation learning methods operate by pulling different views of the same sample together while pushing them away from views of different samples. A fundamental challenge is the estimation of the repulsive force, which typically requires large batch sizes, can exhibit high variance, and is susceptible to the curse of dimensionality. We introduce ScoreCLR, a contrastive representation learning method that offers a novel way to estimate this repulsion term. We consider the framework of representation learning by maximizing the mutual information between two latent views of a sample. By deriving the gradient of the Shannon entropy with respect to the encoder parameters, we obtain an equivalent objective that involves the score of the latent representation. We then use denoising score matching techniques to estimate this score with a lightweight network that is trained jointly with the encoder. We further propose regularization techniques using different noise levels and a Gaussian prior on the latent representation. ScoreCLR replaces the batch of explicit negatives of contrastive methods with a learned, amortized repulsion whose cost is linear in the batch size rather than quadratic. On CIFAR-10 and ImageNet-100 it reaches linear-probe accuracy within 2–4 points of SimCLR under a shared recipe, while producing projector outputs of higher effective rank.

1 INTRODUCTION

The goal of representation learning is to learn features from data that make downstream tasks easier to solve (Goodfellow et al., 2016). It is now widely used across several machine learning domains, including generative modeling (Rombach et al., 2022), reinforcement learning (Hafner et al., 2024), and causality (Schölkopf et al., 2021). Representation learning methods can be broadly divided into two categories: reconstructive approaches (Vincent et al., 2010; Kingma & Welling, 2013; He et al., 2021), which train by reconstructing samples from their latent encodings, and non-reconstructive ones (Chen et al., 2020; Caron et al., 2021; Assran et al., 2023), which optimize a training criterion in the latent space.

For the latter, two main approaches are typically considered. Contrastive methods (van den Oord et al., 2018; Chen et al., 2020) bring representations of different views of the same sample closer while separating those of different samples. Estimating this repulsion force is generally challenging: it necessitates large batch sizes, suffers from high variance and bias (Poole et al., 2019), and requires careful hyperparameter tuning (Chen et al., 2021). This has led to the development of non-contrastive methods that focus solely on aligning latent representations and prevent collapse through architectural design and training asymmetry (Grill et al., 2020; Chen & He, 2021; Caron et al., 2021; Assran et al., 2023) or explicit regularization in the training objective (Bardes et al., 2022; Balestriero & LeCun, 2025). Although these techniques do prevent collapse, it is unclear which structure they impose on the latent space, as well as the efficiency of the estimator they rely on (see Section 3.4).

We introduce ScoreCLR, a novel method for contrastive representation learning that estimates this repulsion term using denoising score matching techniques (Vincent, 2011; Song & Ermon, 2019). Our training criterion derives from maximizing the mutual information between two views of a sample (Tschannen et al., 2020). By deriving the gradient of the Shannon entropy with respect to the encoder parameters, we construct an equivalent loss, sharing the same gradient, that involves

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the score of the latent representation $\nabla_z \log p_Z(z)$. We then learn this score across different noise levels using a lightweight score network trained jointly with the encoder following the methodology of Yin et al. (2024a).

Unlike previous contrastive methods whose loss estimation is quadratic in the batch size, ours is linear: the repulsion is amortized in a score network instead of being computed against the batch. Empirically, ScoreCLR reaches linear-probe accuracy within 2–4 points of SimCLR on CIFAR-10 and ImageNet-100 under a shared recipe, while yielding projector outputs of higher effective rank than SimCLR and LeJEPA, which span only a fraction of the dimensions.

2 Representation learning via mutual information maximization

Let X denote the random variable representing the data. We consider learning representations by learning to distinguish the joint distribution of two different latent views of X from the product of their marginals. More formally, let T denote a stochastic transformation of the data (e.g., cropping, masking, blurring, flipping, etc.) and T_1, T_2 two i.i.d. copies of T. We will consider the latent representation to be $Z = G(T(X))$, where G is an encoding function and denote $Z_1 = G(T_1(X))$ and $Z_2 = G(T_2(X))$ two latent views of X. The objective will be to maximize

$$\mathrm{D}(p_{(Z_1,Z_2)}||p_{Z_1}p_{Z_2}),$$

where D is a divergence criterion. Several divergences have been explored in the literature for learning representations: Wasserstein (Ozair et al., 2019), Maximum Mean Discrepancy (MMD) (Gretton et al., 2005; Li et al., 2021), Jensen–Shannon (Hjelm et al., 2019); in our case we will consider the Kullback–Leibler (KL) divergence that inherits from desirable statistical properties (Cover & Thomas, 2006). We are thus in the InfoMax principle (Linsker, 1988) framework and our objective will be to maximize

$$I(Z_1, Z_2) = \mathrm{KL}(p_{(Z_1,Z_2)}||p_{Z_1}p_{Z_2}) = H(Z_1) - H(Z_1 \mid Z_2).$$

Using this decomposition, we can interpret the objective as simultaneously encouraging the representation to have high entropy and making it predictable from the other representation view, through minimizing the conditional entropy $H(Z_1 \mid Z_2)$. We will parametrize the encoding function with a deep neural network G^φ, denote $Z_1^\varphi = G^\varphi(T_1(X))$, $Z_2^\varphi = G^\varphi(T_2(X))$, and aim at minimizing

$$\mathcal{L}_{\mathrm{MI}}(\varphi) = H(Z_1^\varphi \mid Z_2^\varphi) - H(Z_1^\varphi).$$

We will now present different ways to estimate the mutual information loss that can be found in the literature.

2.1 Variational bound

A common way to construct an estimator is to consider a lower bound using a dual representation of the KL. A standard choice is the Donsker–Varadhan representation, where the supremum is taken over measurable functions f such that both expectations below are finite, which gives us

$$\mathrm{KL}(p_{(Z_1,Z_2)}||p_{Z_1}p_{Z_2}) = \sup_f \left\{ \mathbb{E}\left[f\left(Z_1, Z_2\right)\right] - \log\left(\mathbb{E}\left[e^{f\left(Z_1, \tilde{Z}_2\right)}\right]\right) \right\},$$

where $\tilde{Z}_2 = G^\varphi(\tilde{X})$ with $\tilde{X}$ independent from X. Any f thus gives a lower bound on the mutual information.

2.2 Kernel density estimator

Considering the critic $f_\tau(x, y) = \frac{1}{\tau}\frac{xy^T}{\|x\|\|y\|}$, and using a Monte Carlo estimator for the expectation with i.i.d. samples $(Z_1^i, Z_2^i)_{i\in\{1,\cdots,B\}}$ of (Z_1, Z_2) and $(\tilde{Z}_2^i)_{i\in\{1,\cdots,B\}}$ of $\tilde{Z}_2$, we recover the InfoNCE lower bound of van den Oord et al. (2018)

$$\mathcal{L}_{\text{InfoNCE}} = \underbrace{-\frac{1}{B}\sum_{i=1}^{B} f_\tau(Z_1^i, Z_2^i)}_{\text{attraction}} + \underbrace{\frac{1}{B}\sum_{i=1}^{B} \log\left(\frac{1}{B}\sum_{j=1}^{B}\exp\Big(f_\tau(Z_1^i, \tilde{Z}_2^j)\Big)\right)}_{\text{repulsion}}.$$

The repulsion term is minus a plugin kernel density estimator of the entropy of Z using the kernel $k_\tau(x, y) = \exp(f_\tau(x, y))$. It encourages the latent representation to have high entropy, aiming at uniformity on the sphere (Wang & Isola, 2020).

By symmetrizing the repulsion term with respect to the views, we then recover the SimCLR loss (Chen et al., 2020). Other techniques have been developed to improve this estimator, for example using k-nearest-neighbor techniques (Dwibedi et al., 2021), which relates to the kNN kernel density estimator for the latent (Loftsgaarden & Quesenberry, 1965). Another direction consists in improving the batch sampling technique by considering a dynamic memory queue (He et al., 2020), which enables the use of smaller batch sizes.

These estimators all suffer from the curse of dimensionality under standard regularity conditions on the density of interest (Scott & Wand, 1991; LeCun, 2022). The relevant dimension is in general the intrinsic dimension of the data rather than the ambient one, but it has been recently suggested that this intrinsic dimension is still too large for standard statistical estimators on real-world data (Pope et al., 2021). Contrary to classic kernel density estimators, which only leverage a weak prior on the distribution of interest, the ScoreCLR repulsion estimator includes a neural prior that may benefit from a better sample complexity (Barron, 1993).

2.3 Neural methods

Several approaches have been proposed to estimate or maximize mutual information in representation learning relying on neural critics. For example, MINE (Belghazi et al., 2018) used the dual representation of the KL with a critic learned by a neural network f_ψ, which was subsequently used for unsupervised representation learning in Deep InfoMax (Hjelm et al., 2019), considering local-to-global dependence learning rather than global-to-global as in SimCLR. The mutual-information-based loss can be rewritten as

$$\mathcal{L}(\varphi) = \sup_{\psi} \left\{ \mathbb{E}\left[f_\psi\left(Z_1^\varphi, Z_2^\varphi\right)\right] - \log \mathbb{E}\left[e^{f_\psi\left(Z_1^\varphi, \tilde{Z}_2^\varphi\right)}\right] \right\},$$

where $\tilde{Z}_2^\varphi = G^\varphi(T_2(\tilde{X}))$ with $\tilde{X}$ independent from X.

Training then proceeds by alternating maximization over the critic parameters ψ and maximization of the resulting bound over the encoder parameters φ. An estimator of the mutual information during training is given by

$$\widehat{I}_{\mathrm{DV}}(\varphi, \psi) = \frac{1}{B}\sum_{i=1}^{B} f_\psi\left(Z_{1,i}^\varphi, Z_{2,i}^\varphi\right) - \log\left(\frac{1}{B}\sum_{i=1}^{B} e^{f_\psi\left(Z_{1,i}^\varphi, \tilde{Z}_{2,i}^\varphi\right)}\right).$$

In practice, these variational estimators can suffer from optimization instability and may contribute to mode collapse (Chae et al., 2023). Moreover, these neural estimators do not necessarily preserve desirable properties of the underlying KL divergence, such as invariance to reparameterization (Song & Ermon, 2020). Also, a desirable property of the latent space is that one view is linearly predictive of the other (van den Oord et al., 2018), which is precisely what the cosine-similarity critic rewards. It is thus unclear why it would be beneficial to have a sophisticated critic that learns non-linear relationships between the latents, and simpler critics were indeed found to give better downstream accuracy (Tschannen et al., 2020).

3 ScoreCLR

More recently, diffusion-based approaches have been proposed for mutual information estimation, such as MINDE (Franzese et al., 2024). These methods achieve improved performance and pass the invariance to reparameterization test (Kholkin et al., 2025). To the best of our knowledge, these diffusion-based approaches to mutual information estimation have not yet been explored in a self-supervised representation learning framework.

3.1 EQUIVALENT LOSS

Since we optimize our loss with gradient descent, replacing it by another loss with the same gradient does not change the training dynamic. The following proposition gives such a replacement for the entropy term.

Proposition 3.1. *For any random variable Y, denoting $Z^\varphi = G^\varphi(Y)$, under Assumption A.1 (Appendix A) we have*

$$\nabla_\varphi H(Z^\varphi) = \frac{1}{2}\nabla_\varphi \mathbb{E}[\|Z^\varphi - sg(Z^\varphi + \nabla_z \log p_{Z^\varphi}(Z^\varphi))\|^2],$$

where sg denotes the stop grad operator.

For the conditional entropy, we will consider the Gaussian conditional model $Z_1 \mid Z_2 \sim \mathcal{N}(Z_2, \sigma^2 I)$ and our mutual information loss will then be,

$$\mathcal{L}_{\text{MI}} = \frac{1}{2\sigma^2}\mathbb{E}[\|G^\varphi(T_1(X)) - G^\varphi(T_2(X))\|^2] - H(Z^\varphi)$$

which will give the same training dynamic as

$$\mathcal{L}_{\text{MI-score}}(\varphi) = \frac{1}{2\sigma^2}\mathbb{E}[\|G^\varphi(T_1(X)) - G^\varphi(T_2(X))\|^2] + \frac{1}{2}\mathbb{E}[\|Z^\varphi - \text{sg}(Z^\varphi - \nabla_z \log p_{Z^\varphi}(Z^\varphi))\|^2].$$

This loss can be seen as enforcing invariance of the latent representation to the given transformation while encouraging it to remain of high entropy.

3.2 REGULARIZATION

Gaussian prior Without any extra regularization the problem is only constrained by the architecture of the network and the entropy of the latent could grow arbitrarily high. To alleviate this issue we consider a Gaussian prior on our latent variable by adding an L^2 penalty term to the loss. Noticing that $-H(Z) + \frac{1}{2}\mathbb{E}[\|Z\|^2] = C + \text{KL}(Z \,\|\, \mathcal{N}(0, I))$, we recover the same regularization term as in Balestriero & LeCun (2025), the maximum-entropy prior at fixed variance. In practice the penalty is applied to the noised latent $\bar{Z}_t^\varphi$ at which the score is queried (Algorithm 1).

Noise regularization To estimate the score of the latent we use denoising score matching (Vincent, 2011; Song & Ermon, 2019): we approximate the score of the latent distribution convolved with Gaussian noise of standard deviation t. We denote $Z_t = Z + t\varepsilon$, where $\varepsilon \sim \mathcal{N}(0, I)$, and train a lightweight denoiser D^θ with the conditional denoising loss

$$\mathcal{L}_{\text{CDSM}}(\theta) = \mathbb{E}\big[w(t)\,\|D^\theta(Z_t, t, c) - Z\|^2\big], \qquad \ln t \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2),$$

where c is a conditioning variable (see below) and $w(t)$ is the weighting of Karras et al. (2022). The score is recovered by Tweedie's formula, $s_t^\theta(z, c) = (D^\theta(z, t, c) - z)/t^2$. Further details can be found in Appendix C.

To have a low variance estimate of the repulsive term, we average the score at different noise levels, using independent Gaussian noise for each level; we then get

$$\frac{1}{2}\mathbb{E}\Big[\big\|Z^\varphi - \text{sg}\big(Z^\varphi - \frac{1}{3}\sum_{j=1}^{3} s_{t_j}^\theta(Z_{t_j}^\varphi, c)\big)\big\|^2\Big],$$

with t_1, t_2, t_3 drawn independently and c a conditioning variable.

Conditioning of the score network We condition the score network on a second latent c. During denoising training, c is the other view of the same image, which removes the augmentation variability and makes the denoiser easier to train. During the repulsion query, c is the latent of another image of the batch under the same transformation, so that a view is never repelled from a density estimate built on its own image. Half of the time c is replaced by a learned null embedding, as in classifier-free guidance (Ho & Salimans, 2022).

3.3 TRAINING

Ideally, we would want, for every update of φ, a statistical estimator of the score $s_t^{\theta^*(\varphi)}$ that minimizes $\mathcal{L}_{\text{CDSM}}(\theta)$. This is impossible in practice since it would result in training a diffusion model for every update of φ. Instead we consider the methodology of DMD2 (Yin et al., 2024a), which alternates training between updates of φ and θ with a 1-to-k proportion rule that has proven to be efficient in large-scale training.

For flexibility, we consider tunable weights w_{attr} and w_{rep} between the attraction and the repulsion terms in place of the $1/(2\sigma^2)$ factor of $\mathcal{L}_{\text{MI-score}}$; w_{attr} thus absorbs the constants, and the attraction term is written without the $\frac{1}{2}$ in Algorithm 1. A connection to drifting generative models is given in Appendix B.

Algorithm 1 ScoreCLR Training Loss

Require: Encoder G^φ, denoiser D^θ, augmentations T_1, T_2, noise distribution $p(t)$, score-network steps k, weights $w_{\text{attr}}, w_{\text{rep}}$, conditioning c

$Z_1^\varphi = G^\varphi(T_1(X)), \quad Z_2^\varphi = G^\varphi(T_2(X)), \quad Z^\varphi = [Z_1^\varphi; Z_2^\varphi]$

$\theta \leftarrow k$ updates of $\mathcal{L}_{\text{CDSM}}(\theta)$ on $\text{sg}(Z^\varphi)$ with conditioning c (other view of the same image)

$t_1, t_2, t_3 \sim p(t), \quad \varepsilon_1, \varepsilon_2, \varepsilon_3 \sim \mathcal{N}(0, I), \quad Z_{t_j}^\varphi = Z^\varphi + t_j \varepsilon_j$

$\bar{s} = \text{sg}\big(\frac{1}{3}\sum_{j=1}^3 s_{t_j}^\theta(Z_{t_j}^\varphi, c)\big)$ with c from another image, $\quad \bar{Z}_t^\varphi = \frac{1}{3}\sum_{j=1}^3 Z_{t_j}^\varphi$

$\mathcal{L}_{\text{inv}} = \|Z_1^\varphi - Z_2^\varphi\|^2$

$\mathcal{L}_{\text{max-ent-drift}} = \frac{1}{2}\|Z^\varphi - \text{sg}(Z^\varphi - \bar{s})\|^2$

$\mathcal{L}_{\text{reg}} = \frac{1}{2}\|\bar{Z}_t^\varphi\|^2$

$\mathcal{L}_{\text{ScoreCLR}} = w_{\text{attr}}\mathcal{L}_{\text{inv}} + w_{\text{rep}}\big(\mathcal{L}_{\text{max-ent-drift}} + \mathcal{L}_{\text{reg}}\big)$

$\varphi \leftarrow 1$ update of $\mathcal{L}_{\text{ScoreCLR}}$

3.4 COMPARISON WITH OTHER REGULARIZATION METHODS

Several explicit regularization terms have been proposed to prevent collapse in non-contrastive self-supervised learning. VICReg (Bardes et al., 2022) regularizes the variance and the covariance of the latent, which controls its entropy only through second-order moments: it maximizes the entropy $\frac{1}{2}\log\det\Sigma$ of a Gaussian fit of the latent, and thus implicitly assumes that the latent is Gaussian. SIGReg (Balestriero & LeCun, 2025) pushes the latent toward an isotropic Gaussian with an Epps–Pulley characteristic-function test on random one-dimensional projections, relying on the Cramér–Wold theorem. As for every sliced divergence, the sample complexity of this test does not depend on the dimension (Nadjahi et al., 2020), but its statistical power does: most one-dimensional projections of a high-dimensional distribution are close to Gaussian (Diaconis & Freedman, 1984), and the fraction of directions in which a projection deviates from Gaussian by more than ϵ decays as $\exp(-c\,d\,\epsilon^2)$ (Meckes, 2009), so the number of random slices needed to detect such a deviation grows exponentially with the dimension. A latent supported on a low-dimensional subspace can thus pass the test, the geometric collapse we observe in Section 4.3.

ScoreCLR instead derives its repulsion from the contrastive criterion itself, as the gradient of the entropy term of the mutual information estimated with the full score of the latent, and does not suffer from this geometric collapse (Section 4.3).

4 EXPERIMENTS

We compare ScoreCLR with SimCLR, and with LeJEPA (Balestriero & LeCun, 2025) on CIFAR-10, on linear evaluation, representation geometry, out-of-domain transfer and training cost. Every result is a single run (seed 5), in fp32 except SimCLR (LARS), trained in 16-bit mixed precision as in solo-learn (da Costa et al., 2022). ScoreCLR uses attraction weight 5 on CIFAR-10, and 10 score-network steps with attraction weight 10 on ImageNet-100 (Appendix C); the ablations of Table 6 start from the shared defaults.

4.1 SETUP

Data. We pre-train on CIFAR-10 (Krizhevsky, 2009) and ImageNet-100 (Deng et al., 2009). Transfer targets are CIFAR-10, STL-10 (Coates et al., 2011), and DTD (Cimpoi et al., 2014).

Models. ResNet-18 with a two-layer MLP projector (2048 hidden units, 256 outputs, no BatchNorm: a BatchNorm layer improves the results but partly prevents collapse on its own (Richemond et al., 2020), so we remove it from every method). SimCLR, LeJEPA and ScoreCLR share augmentations, encoder, projector and optimizer (AdamW, Table 7). We also report SimCLR with LARS under solo-learn's recipe (da Costa et al., 2022), with the shared 2048/256 head and its published 4096/512 head.

Evaluation. (i) Offline linear probe and weighted k-NN ($k = 20$) on frozen features; (ii) alignment and uniformity (Wang & Isola, 2020); (iii) out-of-domain linear transfer following Kornblith et al. (2019); (iv) analytical FLOPs and memory per iteration.

4.2 LINEAR EVALUATION

Table 1: Linear-probe and weighted k-NN ($k = 20$) accuracy (%) on frozen features, ResNet-18. Top: shared optimizer and projector. Bottom: SimCLR under solo-learn's LARS recipe with the shared and the published projector (our re-training; offline probe 77.02 against 77.64 online for the published checkpoint).

			CIFAR-10				ImageNet-100			
			Linear		k-NN		Linear		k-NN	
Method	Optim.	Projector	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
SimCLR	AdamW	2048/256	89.45	99.70	88.56	98.84	72.60	92.58	65.06	86.98
LeJEPA	AdamW	2048/256	90.07	99.62	88.35	98.54	–	–	–	–
ScoreCLR (ours)	AdamW	2048/256	87.31	99.43	86.75	98.63	68.88	89.86	63.64	84.68
SimCLR	LARS	2048/256	90.79	99.74	89.99	99.02	75.40	93.30	69.98	88.64
SimCLR	LARS	4096/512	–	–	–	–	77.02	93.84	71.60	89.10

4.3 REPRESENTATION GEOMETRY

Table 2 reports alignment and uniformity of the projector output and Table 3 the effective rank (Garrido et al., 2023) of the backbone features h and of the projector output z. No method collapses in h. In z, SimCLR spans about half of the 256 dimensions on ImageNet-100 and a third on CIFAR-10, the dimensional collapse of InfoNCE described by Jing et al. (2022), and LeJEPA has the lowest rank in both h (266/512) and z (59/256) although its output passes the SIGReg test: the sliced criterion constrains the marginals, not the rank (Section 3.4). ScoreCLR's z has a higher effective rank (220/256 and 191/256) while h stays comparable to SimCLR's. The main recipes are however not close to the $\mathcal{N}(0, I)$ target (uniformity in Table 2). Table 4 makes this precise: with attraction weight 1, ScoreCLR's z is the only one close to $\mathcal{N}(0, I)$ on every statistic (excess kurtosis 0.6, Epps–Pulley statistic 4.6 after whitening against 1.1 for a Gaussian sample of the same size, rank 233/256); LeJEPA has nearly Gaussian projections after whitening (8.7) but on 59 dimensions, and SimCLR fails on both tails and projections. The larger attraction weight of the main recipes trades this Gaussianity for alignment (kurtosis 72.7 at weight 5). A higher rank is not by itself a proxy for accuracy: on the backbone features used by the probe, a lower intrinsic dimension correlates with higher linear-probe accuracy across SSL methods (Mordacq et al., 2026); our claim only concerns z.

4.4 TRANSFER

Table 5 probes frozen features on unseen datasets, STL-10 (near) and DTD (far). ScoreCLR trails SimCLR on both, and the attraction weight is not the cause: with weight 1 instead of 10 it transfers at 68.46 / 80.99 / 52.77.

Table 2: Alignment (↓) and uniformity (↓) (Wang & Isola, 2020) of the projector output, ResNet-18.

		CIFAR-10		ImageNet-100	
Method	Optim.	Align	Unif	Align	Unif
SimCLR	LARS	0.349	-3.818	0.240	-3.856
SimCLR	AdamW	0.355	-3.803	0.236	-3.853
LeJEPA	AdamW	0.313	-3.539	–	–
ScoreCLR (ours)	AdamW	0.438	-3.472	0.174	-2.792
ScoreCLR (ours), default recipe	AdamW	–	–	0.437	-3.877

Table 3: Effective rank (RankMe, Garrido et al., 2023) of the backbone features h and of the projector output z on raw (un-normalised) validation features, out of the space dimension; dimensional collapse of z under InfoNCE is the phenomenon analysed by Jing et al. (2022).

Pre-training	Method	Optim.	h	z
CIFAR-10, ResNet-18	SimCLR	LARS	372 / 512	88 / 256
	SimCLR	AdamW	356 / 512	96 / 256
	LeJEPA	AdamW	266 / 512	59 / 256
	ScoreCLR (ours)	AdamW	375 / 512	220 / 256
ImageNet-100, ResNet-18	SimCLR	LARS	368 / 512	118 / 256
	SimCLR	AdamW	358 / 512	143 / 256
	ScoreCLR (ours)	AdamW	341 / 512	191 / 256

4.5 Ablations

Table 6 isolates the two levers of the recipe. The attraction weight trades uniformity for alignment: weight 5 gains 1.3 points on CIFAR-10 but loses 1.3 on ImageNet-100 and multiplies the loss spikes. Training the score network for 10 steps per iteration instead of 5 is the change that holds on ImageNet-100 ($+1.2$ offline, $+2.3$ k-NN, 6 spikes instead of 17), and with 10 steps a weight of 10 gains a further 0.9 points. The spikes come from repulsion queries at small noise levels; clamping them removes the early spikes but not the late ones and does not change the final accuracy (Appendix C.5).

4.6 Computational Cost

InfoNCE's similarity matrix is quadratic in the batch ($6N^2d$ FLOPs, $16N^2$ bytes for $N = 2B$ codes); ScoreCLR queries a 0.38M-parameter score network per code, linear in B (27 MFLOPs and 30 KB per image). The objective is cheaper beyond $B \approx 4.5$k in FLOPs and $B \approx 500$ in memory (Figure 1), but in a full iteration the encoder dominates: at $B \leq 4096$ the two methods cost the same within 0.3% FLOPs and 4% memory, and the per-device memory saving under data parallelism reaches 27% only for ResNet-18 at 32 px and $B = 32768$. FLOPs do not capture latency either: with 5 score-network steps, ScoreCLR takes 0.30 min per epoch on CIFAR-10 (one H100) against 0.18 for SimCLR under the shared fp32 recipe ($1.7\times$), and 1.07 against 0.93 on ImageNet-100 (two H100s; SimCLR under its published LARS recipe in 16-bit), the 10-step recipe being a further $1.25\times$ slower. ScoreCLR is thus slower in practice at the batch sizes we use, and it requires fp32 training.

5 Conclusion

We introduced ScoreCLR, which replaces the batch-based repulsion of contrastive learning by a score-based estimate of the entropy gradient, learned jointly with the encoder. A natural extension is a fully score-based critic, where the conditional entropy is also derived from the learned score. Another direction is to reduce the cost of the method by lowering the number of score-network steps needed per encoder update. Finally, since the score-based estimator keeps the projector output at a higher effective rank than existing regularizers (Section 4.3), it would be interesting to study priors

Table 4: Normality of the projector output z on raw validation features ($d = 256$; 10 000 images on CIFAR-10, 5 000 on ImageNet-100). Kurtosis: mean per-coordinate excess kurtosis. EP: sliced Epps–Pulley statistic of SIGReg (256 random directions, 17 knots) on ZCA-whitened z; an i.i.d. $\mathcal{N}(0, I)$ sample of the same size gives 1.1 ± 0.1. Top-1: share of the variance on the first principal component.

Pre-training	Method	Kurtosis	EP	RankMe z	Top-1 (%)
CIFAR-10	SimCLR (LARS)	2.72	921	88	2.7
	LeJEPA	0.36	8.7	59	5.5
	ScoreCLR, attraction weight 1	1.15	14.9	238	3.5
ImageNet-100	SimCLR (AdamW)	3.66	80.9	143	3.1
	ScoreCLR, attraction weight 1	0.61	4.6	233	3.6
	ScoreCLR, attraction weight 5, 10 steps	72.7	16.5	190	9.7

Table 5: Out-of-domain linear transfer, top-1 (%): logistic regression on frozen L2-normalized features at the pretraining resolution (Appendix C.4). "·" is the pretraining dataset (Table 1).

Method	Optim.	CIFAR-10	STL-10	DTD
Pretrained on CIFAR-10 (ResNet-18, 32 px)				
SimCLR	AdamW	·	76.05	30.96
LeJEPA	AdamW	·	75.66	26.38
ScoreCLR (ours)	AdamW	·	73.91	26.81
Pretrained on ImageNet-100 (ResNet-18, 224 px)				
SimCLR	AdamW	76.46	84.21	60.16
ScoreCLR (ours)	AdamW	67.92	80.36	53.72
ScoreCLR (ours), attraction weight 1	AdamW	68.46	80.99	52.77

with finer geometric properties than the isotropic Gaussian and to compare with those methods in that setting.

Reproducibility statement

The assumptions and the full proof of Proposition 3.1 are in Appendix A. Every training recipe (architecture, optimizer, schedule, augmentations, score-network settings and loss weights) is listed in Appendix C and Table 7; the linear-probe, k-NN, effective-rank and transfer protocols are in Appendices C.2–C.4, the ablation and spike definitions in Appendix C.5, and the cost model in Appendix C.6. All results are single runs with seed 5. The implementation builds on the public solo-learn library (da Costa et al., 2022); the code will be released upon publication.

AI use statement

In this work, we used generative AI tools (Claude Code) to implement methods (the ScoreCLR training code and evaluation scripts, on top of the solo-learn library), to design and run experiments (launching and monitoring training and evaluation jobs, and suggesting ablations), to interpret results (drafting the analysis of the experimental tables), and to assist in the writing of proofs. We have not used generative AI tools to generate synthetic data sets, to develop theoretical models or conceptual frameworks, to formulate mathematical claims, to provide critical ingredients for proving mathematical claims, or to propose or refine hypotheses; assisting with translation, cleaning and reformatting datasets and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create scientific figures, to draft parts of the experimental section, to edit the paper for grammar and readability, to identify relevant literature and to format references. We have reviewed all AI-assisted work: the code was verified by the authors against the paper's equations, every number in the tables was checked against the training logs, and all suggested references were read by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Table 6: ScoreCLR ablations, ResNet-18; each row changes one setting of the block's default unless two are listed. Top-1 / k-NN: offline probe and k-NN top-1; Align / Unif on the projector output. Spikes: logged training steps (one every 50) whose repulsion loss exceeds $3\times$ the run median, with the maximum in parentheses. "main" marks the recipe of the main tables; the full set of ablations is in Table 8.

Change	Epochs	Top-1	k-NN	Align	Unif	Spikes
CIFAR-10						
Default	1000	86.04	85.04	0.603	-3.873	33 (51.4)
Attraction weight $1 \to 5$ (main)	1000	87.31	86.75	0.438	-3.472	31 (537)
ImageNet-100						
Default	400	66.74	60.80	0.437	-3.877	17 (1.1×10^4)
Attraction weight $1 \to 5$	400	65.48	59.62	0.276	-3.420	95 (1.2×10^{11})
Score-net steps $5 \to 10$	400	67.98	63.14	0.396	-3.861	6 (48.0)
Steps $5 \to 10$, attraction $1 \to 10$ (main)	400	68.88	63.64	0.174	-2.792	–

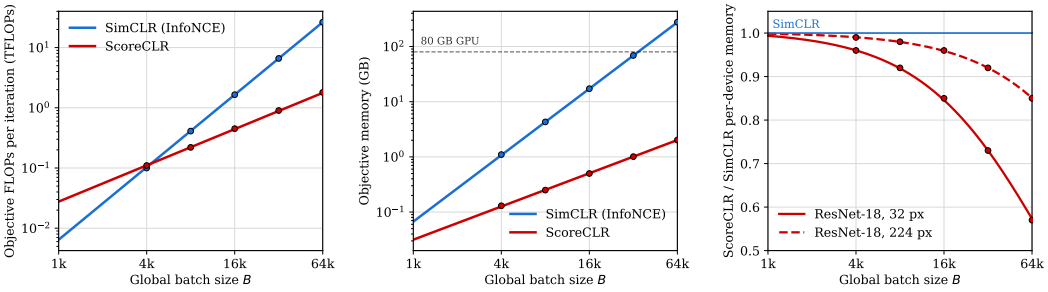

Figure 1: Analytical cost vs. global batch size B (two views per image). Left, centre: FLOPs and memory of the objective alone (ScoreCLR with 5 score-network steps). Right: ScoreCLR / SimCLR per-device memory for a full data-parallel iteration in 16-bit mixed precision; the ratio does not depend on the number of devices. Values are tabulated in Table 9 (Appendix C.6).

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025. doi: 10.48550/arXiv.2511.08544. URL https://arxiv.org/abs/2511.08544.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022.

Andrew R Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R. Devon Hjelm. MINE: Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 531–540. PMLR, 2018.

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.

Jiseok Chae, Kyuwon Kim, and Donghwan Kim. Open problem: Is there a first-order method that only converges to local minimax optima? In *Proceedings of the Thirty-Sixth Conference on*

Learning Theory, volume 195 of *Proceedings of Machine Learning Research*, pp. 5957–5964. PMLR, 2023.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 1597–1607. PMLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple Siamese representation learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 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 (ICCV)*, pp. 9640–9649, 2021.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2 edition, 2006.

Victor Guilherme Turrisi da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23(56):1–6, 2022.

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.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting, 2026. URL `https://arxiv.org/abs/2602.04770`. arXiv preprint.

Persi Diaconis and David Freedman. Asymptotics of graphical projection pursuit. *The Annals of Statistics*, 12(3):793–815, 1984.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9588–9597, 2021.

Giulio Franzese, Mustapha Bounoua, and Pietro Michiardi. Minde: Mutual information neural diffusion estimation. *arXiv preprint arXiv:2310.09031*, 2024. doi: 10.48550/arXiv.2310.09031.

Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann LeCun. RankMe: Assessing the downstream performance of pretrained self-supervised representations by their rank. In *International Conference on Machine Learning (ICML)*, 2023.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Arthur Gretton, Olivier Bousquet, Alexander J. Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International Conference on Algorithmic Learning Theory*, pp. 63–77. Springer, 2005.

Arthur Gretton, Li Kevin Wenliang, Alexandre Galashov, James Thornton, Valentin De Bortoli, and Arnaud Doucet. On the wasserstein gradient flow interpretation of drifting models. *arXiv preprint arXiv:2605.05118*, 2026.

Jean-Bastien Grill, Florian Strub, Florent Altché, 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. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2024. doi: 10.48550/arXiv.2301.04104.

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 (CVPR)*, pp. 9729–9738, 2020. doi: 10.1109/CVPR42600.2020.00975.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2019.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Li Jing, Pascal Vincent, Yann LeCun, and Yuandong Tian. Understanding dimensional collapse in contrastive self-supervised learning. In *International Conference on Learning Representations*, 2022.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 26565–26577, 2022.

Sergei Kholkin, Ivan Butakov, Evgeny Burnaev, Nikita Gushchin, and Alexander Korotin. InfoBridge: Mutual information estimation via bridge matching. *arXiv preprint arXiv:2502.01383*, 2025. doi: 10.48550/arXiv.2502.01383.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do better ImageNet models transfer better? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann LeCun. A path towards autonomous machine intelligence, version 0.9.2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Yazhe Li, Roman Pogodin, Danica J. Sutherland, and Arthur Gretton. Self-supervised learning with kernel dependence maximization. In *Advances in Neural Information Processing Systems*, volume 34, pp. 15543–15556, 2021.

Ralph Linsker. Self-organization in a perceptual network. *Computer*, 21(3):105–117, 1988.

D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, 1965. doi: 10.1214/aoms/1177700079.

Elizabeth Meckes. Quantitative asymptotics of graphical projection pursuit. *Electronic Communications in Probability*, 14:176–185, 2009.

Julie Mordacq, Vicky Kalogeiton, and Steve Oudot. IdEst: Assessing self-supervised learning representations via intrinsic dimension. *arXiv preprint arXiv:2606.03338*, 2026.

Kimia Nadjahi, Alain Durmus, Lénaïc Chizat, Soheil Kolouri, Shahin Shahrampour, and Umut Simsekli. Statistical and topological properties of sliced probability divergences. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. *arXiv preprint arXiv:1903.11780*, 2019. doi: 10.48550/arXiv.1903.11780. URL https://arxiv.org/abs/1903.11780.

Ben Poole, Sherjil Ozair, Aaron van den Oord, Alexander A. Alemi, and George Tucker. On variational bounds of mutual information. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 5171–5180. PMLR, 2019.

Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Pierre H. Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, and Michal Valko. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2022.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning. *arXiv preprint arXiv:2102.11107*, 2021. doi: 10.48550/arXiv.2102.11107.

David W. Scott and M. P. Wand. Feasibility of multivariate density estimates. *Biometrika*, 78(1): 197–205, 1991.

Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv preprint arXiv:1910.06222*, 2020. doi: 10.48550/arXiv.1910.06222. URL https://arxiv.org/abs/1910.06222.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pp. 11895–11907, 2019.

Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://arxiv.org/abs/1907.13625.

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

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010. URL https://jmlr.csail.mit.edu/papers/volume11/vincent10a/vincent10a.pdf. Editor: Léon Bottou.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 9929–9939. PMLR, 2020. URL https://arxiv.org/abs/2005.10242.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a. doi: 10.48550/arXiv.2405.14867.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024b. doi: 10.48550/arXiv.2311.18828.

A PROOFS

Assumption A.1.

(i) For every x, the map $\varphi \mapsto G^\varphi(x)$ is continuously differentiable.

(ii) Z^φ has a positive density p_{Z^φ}, continuously differentiable in (z, φ), with finite entropy, finite second moment and finite Fisher information $\mathbb{E}\big[\|\nabla_z \log p_{Z^\varphi}(Z^\varphi)\|^2\big]$.

(iii) Derivatives with respect to φ can be exchanged with expectations and integrals (e.g. by dominated convergence).

Proof of Proposition 3.1. Since $Z^\varphi = G^\varphi(Y)$, the law of the unconscious statistician gives

$$H(Z^\varphi) = -\,\mathbb{E}_{Z^\varphi}\big[\log p_{Z^\varphi}(Z^\varphi)\big] = -\,\mathbb{E}_Y\big[\log p_{Z^\varphi}(G^\varphi(Y))\big].$$

The encoder parameters enter in two places: through the point $G^\varphi(Y)$ at which the density is evaluated, and through the density p_{Z^φ} itself. Differentiating both,

$$\nabla_\varphi H(Z^\varphi) = -\,\mathbb{E}_Y\Big[\big(\nabla_\varphi G^\varphi(Y)\big)^\top \big(\nabla_z \log p_{Z^\varphi}\big)(G^\varphi(Y))\Big] - \,\mathbb{E}_Y\Big[\big(\nabla_\varphi \log p_{Z^\varphi}\big)(G^\varphi(Y))\Big],$$

where in the second term ∇_φ acts on the parameter index of the density only, the evaluation point being held fixed.

The second term vanishes. Indeed,

$$\begin{aligned}\mathbb{E}_Y\big[\big(\nabla_\varphi \log p_{Z^\varphi}\big)(G^\varphi(Y))\big] = \mathbb{E}_{Z^\varphi}\big[\nabla_\varphi \log p_{Z^\varphi}(Z^\varphi)\big] &= \int p_{Z^\varphi}(z)\,\frac{\nabla_\varphi p_{Z^\varphi}(z)}{p_{Z^\varphi}(z)}\,\mathrm{d}z \\ &= \nabla_\varphi \int p_{Z^\varphi}(z)\,\mathrm{d}z = \nabla_\varphi(1) = 0,\end{aligned}$$

the interchange of ∇_φ and the integral being justified by dominated convergence. Hence

$$\nabla_\varphi H(Z^\varphi) = -\,\mathbb{E}_Y\Big[\big(\nabla_\varphi G^\varphi(Y)\big)^\top \big(\nabla_z \log p_{Z^\varphi}\big)(G^\varphi(Y))\Big]. \tag{1}$$

It remains to identify the right-hand side as the gradient of the stated surrogate. Write

$$\mathcal{L}(\varphi) = \tfrac{1}{2}\,\mathbb{E}_Y\big[\|G^\varphi(Y) - c^\varphi(Y)\|^2\big], \qquad c^\varphi(Y) := \mathrm{sg}\Big(G^\varphi(Y) + \big(\nabla_z \log p_{Z^\varphi}\big)(G^\varphi(Y))\Big).$$

The stop-gradient has zero derivative, so $\nabla_\varphi c^\varphi(Y) = 0$ and only the first occurrence of G^φ is differentiated:

$$\nabla_\varphi \mathcal{L}(\varphi) = \mathbb{E}_Y\Big[\big(\nabla_\varphi G^\varphi(Y)\big)^\top \big(G^\varphi(Y) - c^\varphi(Y)\big)\Big].$$

At the point of evaluation the stop-gradient is the identity, so $G^\varphi(Y) - c^\varphi(Y) = -\big(\nabla_z \log p_{Z^\varphi}\big)(G^\varphi(Y))$, and therefore

$$\nabla_\varphi \mathcal{L}(\varphi) = -\,\mathbb{E}_Y\Big[\big(\nabla_\varphi G^\varphi(Y)\big)^\top \big(\nabla_z \log p_{Z^\varphi}\big)(G^\varphi(Y))\Big] = \nabla_\varphi H(Z^\varphi)$$

by equation 1. □

B Connection to drifting models

Considering the terminology of Deng et al. (2026), we denote as drifting field

$$\mathrm{drift}_i^{\varphi} = -\nabla_z \log p_{Z^{\varphi}}(Z_i^{\varphi}) - Z_i^{\varphi}, \qquad i \in \{1, 2\},$$

the velocity field of the Wasserstein gradient flow of $KL(\cdot \,\|\, \mathcal{N}(0, I))$ (Gretton et al., 2026). Symmetrizing the repulsion force with respect to the views and using the stop-gradient trick of DMD (Yin et al., 2024b), we get the mutual information loss with Gaussian prior, which writes as

$$\mathcal{L}_{\text{MI-score}}^{\text{Gaussian prior}}(\varphi) = \lambda\, \mathbb{E}\big[\|Z_1^{\varphi} - Z_2^{\varphi}\|^2\big] + \frac{1}{2}\sum_{i=1}^{2} \mathbb{E}\big[\|Z_i^{\varphi} - \mathrm{sg}(Z_i^{\varphi} + \mathrm{drift}_i^{\varphi})\|^2\big].$$

C Experimental Details

C.1 Pre-training

Table 7 lists every recipe. Under the shared recipe every method uses the same two-layer MLP projector (Linear–ReLU–Linear, no BatchNorm), AdamW (weight decay 10^{-6}, gradient-norm clip 1.0) with an encoder learning rate tuned per method, held constant and decayed to zero by a cosine over the last 100 epochs ($2{\times}10^{-4}$ for SimCLR and LeJEPA; 10^{-4} on CIFAR-10 and $2{\times}10^{-4}$ on ImageNet-100 for ScoreCLR), batch 128 for 1000 epochs on CIFAR-10 and 2×128 for 400 epochs on ImageNet-100, in fp32. SimCLR and ScoreCLR share the augmentation pipeline (random crop, flip, colour jitter 0.8/0.8/0.8/0.2, grayscale, and blur and solarization on ImageNet-100). LeJEPA on CIFAR-10 uses the ScoreCLR pipeline with SIGReg in place of the score-based repulsion ($\lambda = 0.02$, 17 knots, $t_{\max} = 3$, 256 slices): ResNet-18, 2048/256 projector, AdamW $2{\times}10^{-4}$ with the shared schedule, batch 128, 1000 epochs, fp32. The SimCLR (LARS) rows are our own re-training with solo-learn's LARS recipe (lr 0.3, weight decay 10^{-4}, 10-epoch warm-up and cosine, $\tau = 0.2$, 400 epochs, batch 2×128, 16-bit mixed precision) and the projector narrowed from the published 4096/512 to the shared 2048/256 head. The 4096/512 control row re-trains SimCLR with every other hyper-parameter at the published default. ScoreCLR on CIFAR-10 uses an MLP-FiLM score network, encoder AdamW 10^{-4}, score-network LR 10^{-3} ($10\times$ the encoder), attraction weight 5, batch 128, 1000 epochs at constant LR with a cosine cool-down over the last 100, in fp32. ImageNet-100 runs use 400 epochs at 2×128 with a 100-epoch cosine cool-down and DSM noise $p_{\text{std}} = 0.8$. The ImageNet-100 configuration of the main tables trains the score network for 10 steps per iteration instead of 5 and uses attraction weight 10. In the implementation the attraction term is $\|Z_1^{\varphi} - Z_2^{\varphi}\|^2 / t_{\text{attr}}$ with $t_{\text{attr}} = 1$; the ablation of Table 8 varies this scale.

C.2 Linear evaluation

The offline probe trains a linear classifier for 100 epochs on frozen, L2-normalized backbone features with SGD (momentum 0.9, lr 10, warm-up and cosine) in fp32, with the standard crop-and-flip augmentation. L2 normalization is needed because a small fraction of ScoreCLR's ImageNet-100 features has norms two to four orders of magnitude above the median under eval-mode BatchNorm, which dominates the gradients of a raw-feature probe; on spike-free runs the raw and normalized probes agree to within 0.1 points, and SimCLR gains 0.3 to 1.3 points from the same normalization. k-NN uses $k = 20$ cosine neighbours on the same features.

C.3 Effective rank

RankMe (Garrido et al., 2023) is the exponential of the entropy of the normalized singular values of the feature matrix, computed on the validation set. We report it on both the backbone features and the projector output because Jing et al. (2022) (DirectCLR) showed that contrastive learning collapses the projector output while the backbone stays full-rank.

C.4 Out-of-domain transfer

Following Kornblith et al. (2019), a multinomial logistic-regression classifier is fit on frozen, L2-normalized backbone features of each downstream training set. The L2 penalty is selected on a

Table 7: Pre-training hyperparameters (CIFAR-10 / ImageNet-100). The SimCLR column is solo-learn's LARS recipe; under the shared recipe, SimCLR and LeJEPA use the ScoreCLR optimizer settings with encoder LR 2×10^{-4}. ImageNet-100 uses 2×128 images per step with DDP-averaged gradients and no learning-rate rescaling.

	SimCLR (LARS)	ScoreCLR
Backbone	ResNet-18 / ResNet-18	ResNet-18 / ResNet-18
Epochs	1000 / 400	1000 / 400
Batch size	256 / 2×128	128 / 2×128
Optimizer	LARS	AdamW (encoder and score net)
Base LR	0.4 / 0.3	10^{-4} / 2×10^{-4}
Classifier LR	0.1 / 0.1	0.01
Weight decay	10^{-4}	10^{-6} (score net: 10^{-2})
LR schedule	10-epoch warm-up + cosine	constant, cosine over the last 100 ep
Precision	16-bit mixed	fp32
Projector	Linear–ReLU–Linear, no BatchNorm	Linear–ReLU–Linear, no BatchNorm
Hidden / output dim	2048/256 (published: 4096/512)	2048/256
Temperature	$\tau = 0.2$	attraction noise $t_{\text{attr}} = 1.0$
Score network	–	MLP-FiLM, 3 layers $\times$ 256, 64-d time emb.
Score-net LR	–	10^{-3} ($10\times$ / $5\times$ encoder)
Score-net steps per iteration	–	5 (ImageNet-100 main recipe: 10)
DSM noise $\ln t$	–	$\mathcal{N}(-1.2,\ 1.2^2)$ / $\mathcal{N}(-1.2,\ 0.8^2)$
Query samples K	–	3
Conditional repulsion (null prob.)	–	yes (0.5)
Attraction / repulsion weight	–	5.0 / 1.0 (CIFAR-10), 10.0 / 1.0 (ImageNet-100)

validation split (the dataset's own where one exists, else a stratified 20% hold-out) and the classifier is refit on train + val. Downstream images are resized and center-cropped to the pretraining resolution: 32×32 with the CIFAR stem for CIFAR-10 backbones and 224×224 for ImageNet-100 backbones.

C.5 Ablations

Instability comes from repulsion queries at small t: queries with $t < 0.01$ are 0.2% of the draws but 13% of the repulsion loss. Clamping the queries to $t \in [0.1, 2]$ or narrowing the noise distribution (std of $\ln t$ from 1.2 to 0.8, the ImageNet-100 default) removes every early spike (Table 6), and the narrower noise gains 1.3 points online at epoch 250, but late spikes remain: a clamped 1000-epoch run still spikes 8 times after epoch 750 and ends where an unclamped one does (86.50 vs. 86.61 offline). The main recipes therefore keep the queries unclamped. Short-budget gains rarely persist: repulsion weight 0.5 leads by 6.9 points online at epoch 49 and ends at parity. Attraction weight 5 holds at 1000 epochs on CIFAR-10 ($+1.3$ offline, $+1.7$ k-NN, trading uniformity for alignment) but costs 1.3 points on ImageNet-100 and multiplies its spikes. On ImageNet-100 the change that holds at 400 epochs is 10 score-network steps instead of 5 ($+1.2$ offline, $+2.3$ k-NN, 6 spikes instead of 17); with 10 steps, attraction weight 10 gains a further 0.9 points (the main recipe) while 5 is neutral. A 256-unit projector trails by 4.8 points at epoch 200.

Each ablation row changes one setting of its block's default. A spike is a logged training step (one every 50, on both datasets) whose repulsion loss exceeds three times the run's median; the maximum is reported alongside. The constant-LR default is the 500-epoch default read at epochs 200 and 250, so only its online accuracy exists. The repulsion queries reuse the score network's training noise distribution, $\ln t \sim \mathcal{N}(-1.2,\ 1.2^2)$, which is unbounded below; the clamp restricts the queries to $[0.1, 2]$, and the noise-std row narrows both distributions to $\mathcal{N}(-1.2,\ 0.8^2)$. The late-spike comparison in the text uses encoder LR 2×10^{-4} and score-net LR $5\times$: the clamped run cools down over the last 100 epochs, the unclamped one over the last 50.

C.6 Cost model

With $N = 2B$ codes of dimension $d = 256$, InfoNCE's similarity matrix costs $6N^2d$ FLOPs and about $16N^2$ bytes. ScoreCLR queries its score network ($P = 0.38$M parameters) separately for each code with $S = 5$ score-matching updates and $K = 3$ repulsion queries, about $2P(6S + 2K) =$ 27 MFLOPs and 30 KB per image. Architecture constants are counted on the implemented modules. For the full-iteration memory ratio, activations saved for backward are counted in 16-bit mixed precision; under data parallelism over W devices each device holds a $2B/W \times 2B$ block of the

Table 8: Full ScoreCLR ablations, ResNet-18; each row changes one setting of the block's default unless two are listed. Online / Top-1 / k-NN: online classifier, offline probe and k-NN top-1; Align / Unif on the projector output. Spikes: logged training steps (one every 50) whose repulsion loss exceeds $3\times$ the run median, with the maximum in parentheses. "main" marks the recipe of the main tables.

Change	Epochs	Online	Top-1	k-NN	Align	Unif	Spikes
CIFAR-10, constant LR							
Default[†]	200/250	75.24/76.78	–	–	–	–	4 (22.1)
Query t clamped to $[0.1, 2]$	200	74.82	76.91	74.57	0.862	-3.897	0 (3.6)
Noise std $1.2 \to 0.8$ (DSM, queries)	250	78.11	80.40	78.47	0.842	-3.911	0 (3.8)
Attraction t_{attr} $1.0 \to 0.5$	200	77.00	78.07	76.35	0.743	-3.805	4 (41.5)
CIFAR-10, encoder 10^{-4}*, score-net LR* $10\times$*, cosine cool-down over the last 100*							
Default	1000	86.10	86.04	85.04	0.603	-3.873	33 (51.4)
Attraction weight $1 \to 5$ (main)	1000	87.47	87.31	86.75	0.438	-3.472	31 (537)
ImageNet-100							
Default	400	66.10	66.74	60.80	0.437	-3.877	17 (1.1×10^{4})
Attraction weight $1 \to 5$	400	65.22	65.48	59.62	0.276	-3.420	95 (1.2×10^{11})
Score-net LR not decayed in cool-down	400	65.50	66.20	59.74	0.457	-3.894	19 (2.1×10^{3})
Score-net steps per iteration $5 \to 10$	400	67.64	67.98	63.14	0.396	-3.861	6 (48.0)
Steps $5 \to 10$, attraction $1 \to 5$	400	67.44	67.94	62.36	0.260	-3.440	85 (2.1×10^{7})
Steps $5 \to 10$, attraction $1 \to 10$ (main)	400	68.42	68.88	63.64	0.174	-2.792	–
Projector hidden width $2048 \to 256$	200[‡]	53.76	–	–	–	–	1 (8.4)

[†]The 500-epoch default read at epochs 200 and 250; no checkpoint there, so online only. [‡]Stopped at epoch 200 (default: 58.6).

similarity matrix, so the ratio is independent of W. At $B = 32768$ ScoreCLR's objective needs $7\times$ fewer FLOPs and $68\times$ less memory than InfoNCE, and the per-device saving is 27% for ResNet-18 at 32 px and 8% at 224 px; SimCLR's objective alone exceeds one 80 GB GPU at $B = 65536$.

Table 9: Analytical cost vs. global batch size B. Left: the objective alone. Right: ScoreCLR / SimCLR per-device memory for a full data-parallel iteration.

	Objective only				Full iteration, memory ratio	
	TFLOPs		Memory (GB)		R18	R18
B	SimCLR	ScoreCLR	SimCLR	ScoreCLR	32 px	224 px
4096	0.10	0.11	1.1	0.13	0.96	0.99
8192	0.41	0.22	4.3	0.25	0.92	0.98
16384	1.65	0.45	17.2	0.50	0.85	0.96
32768	6.60	0.90	68.9	1.01	0.73	0.92
65536	26.4	1.80	276	2.02	0.57	0.85