

MVEB: Self-Supervised Learning with Multi-View Entropy Bottleneck

Liangjian Wen, Xiasi Wang, Jianzhuang Liu, Zenglin Xu

Abstract—Self-supervised learning aims to learn representation that can be effectively generalized to downstream tasks. Many self-supervised approaches regard two views of an image as both the input and the self-supervised signals, assuming that either view contains the same task-relevant information and the shared information is (approximately) sufficient for predicting downstream tasks. Recent studies show that discarding superfluous information not shared between the views can improve generalization. Hence, the ideal representation is sufficient for downstream tasks and contains minimal superfluous information, termed minimal sufficient representation. One can learn this representation by maximizing the mutual information between the representation and the supervised view while eliminating superfluous information. Nevertheless, the computation of mutual information is notoriously intractable. In this work, we propose an objective termed multi-view entropy bottleneck (MVEB) to learn minimal sufficient representation effectively. MVEB simplifies the minimal sufficient learning to maximizing both the agreement between the embeddings of two views and the differential entropy of the embedding distribution. Our experiments confirm that MVEB significantly improves performance. For example, it achieves top-1 accuracy of 76.9% on ImageNet with a vanilla ResNet-50 backbone on linear evaluation. To the best of our knowledge, this is the new state-of-the-art result with ResNet-50.

Index Terms—Self-supervised learning, Minimal sufficient representation, Representation learning

1 INTRODUCTION

Self-supervised learning (SSL) has achieved significant progress in learning representation to generalize well to broad downstream tasks. Many state-of-the-art SSL approaches maximize the agreement between the embeddings of two views of an image from a multi-view perspective. These works are based on Siamese networks and employ different methods to deal with the collapse problem during representation learning. For example, contrastive learning [1]–[5] utilizes negative samples to separate features of different images to avoid collapse. Asymmetric network methods [6]–[8] introduce a predictor network and a momentum encoder (or a stop-gradient operation) to prevent collapse without negative samples. In addition, feature decorrelation methods [9], [10] avoid collapse by reducing the redundancy among the feature dimensions. Empirical results of these SSL works show competitive performance on multiple visual tasks compared with supervised learning methods.

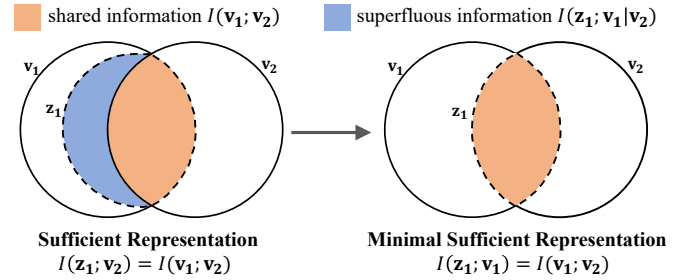


Fig. 1. Illustration of the sufficient and minimal representation in the unsupervised multi-view setting. The common assumption in multi-view learning is that the information $I(v_1; v_2)$ shared between view v_1 and view v_2 is sufficient for the prediction of downstream tasks [11]. When $I(z_1; v_2) = I(v_1; v_2)$, z_1 (denoted by the dotted line in the above figure) contains all the task-relevant information shared between the two views (left). Hence, z_1 is a sufficient representation. If all superfluous information is eliminated, i.e., $I(z_1; v_1) = I(v_1; v_2)$ (right), z_1 is the minimal sufficient representation.

From the multi-view perspective, self-supervised approaches often consider two image views as the input and the self-supervised signals for each other. One can assume that either view is (approximately) sufficient for the prediction of downstream tasks and contains the same task-relevant information in multi-view learning [11]. This suggests that different views of an image should not affect the prediction for downstream tasks. Similar to the role of labels in supervised learning, two views as the mutual self-supervised signals are adapted to extract task-relevant information based on Siamese networks. If the learned representation contains the same task-relevant information shared between the two views, it is sufficient for downstream tasks. Moreover, superfluous information is identified as that not shared by both views. As shown in [12] and [13], discarding superfluous information can improve the generalization of

- Corresponding to Zenglin Xu
- L. Wen is with the School of Computing and Artificial Intelligence, and Research Institute for Digital Economy and Interdisciplinary Sciences, Southwestern University of Finance and Economics, Chengdu, China. E-mail: wlj6816@gmail.com
- X. Wang is with the Hong Kong University of Science and Technology, Hong Kong, China. E-mail: xwangfy@connect.ust.hk
- J. Liu is with Shenzhen Institute of Advanced Technology, Shenzhen, China. E-mail: jz.liu@siaat.ac.cn
- Z. Xu is with the Harbin Institute of Technology Shenzhen, Shenzhen, China, and the Pengcheng Laboratory, Shenzhen, China. E-mail: zenglin@gmail.com
- This work was partially supported by an Open Research Project of Zhejiang Lab (NO.2022RC0AB04), a Major Key Project of PCL (No. PCL2023A09), and a key program of fundamental research from Shenzhen Science and Technology Innovation Commission (No. JCYJ20200109113403826)

the learned representation for downstream tasks. Hence, the ideal representation is sufficient for downstream tasks and contains minimal superfluous information. In light of the information bottleneck principle [14], the minimal sufficient representation is defined in the unsupervised setting, as shown in Fig. 1. One can learn the minimal sufficient representation by minimizing the mutual information between the extracted feature and its input view conditioned on the other supervised view while maximizing the mutual information between the extracted feature and the supervised view. This is termed the multi-view information bottleneck [12]. However, the computation of mutual information is notoriously intractable. Although the variational method [12] can be introduced to overcome the intractability, [15] shows this cannot improve the performance for downstream tasks much compared with SimCLR [2]. Learning the minimal sufficient representation effectively in self-supervised representation learning is still challenging.

To address this problem, we propose a new objective function, the multi-view entropy bottleneck (MVEB), to learn the minimal sufficient representation. Our method can learn task-relevant information and eliminate superfluous information, which is related to the multi-view information bottleneck. MVEB simplifies the minimal sufficient learning to the process of maximizing both the agreement between the embeddings of two views of an image and the differential entropy of the embedding distribution. Moreover, it can be directly applied to Siamese networks without modification of the network structure and other complex designs.

However, it is intractable to compute the differential entropy of the embedding distribution since it is unknown. We propose a score-based entropy estimator with the von Mises-Fisher kernel [16] to approximate the gradient of the differential entropy with model parameters, such that we can directly use the gradient approximation with model parameters for backpropagation to maximize the differential entropy. It can increase the uniformity of the embeddings efficiently. Moreover, this formulation does not require a large batch size or a memory bank.

We empirically demonstrate that MVEB significantly improves the generalization of the learned representation for downstream tasks. Our main contributions are summarized as follows:

- We propose MVEB to learn the minimal sufficient representation in the unsupervised multi-view setting. It can be directly applied to Siamese networks without modification of the network structure and other complex designs.
- We present a score-based entropy estimator with the von Mises-Fisher kernel to approximate the gradient of the differential entropy of the embedding distribution w.r.t. model parameters. This estimator can be used to maximize the differential entropy to increase uniformity.
- We first analyze contrastive learning (e.g., SimCLR and MOCO), asymmetric network methods (e.g., BYOL and SimSiam), and feature decorrelation methods (e.g., Barlow Twins and VICReg) from learning the minimal sufficient representation. Based on

MVEB, we argue that these methods also try to learn the minimal sufficient representation by optimizing alignment and uniformity.

- Comprehensive experiments are conducted to show the superior performance of MVEB. For example, it achieves top-1 accuracy of 76.9% on ImageNet with a vanilla ResNet-50 backbone with a single-layer classifier fine-tuned. To the best of our knowledge, this is the new state-of-the-art result with ResNet-50.

2 RELATED WORK

Self-supervised learning (SSL) learns representation by defining a pretext task without annotation. Pretext tasks are the core of SSL to provide supervision signals to mine the data structure. Recently, contrastive learning [2], [3] makes promising progress and has reduced the gap with supervised learning on many computer vision benchmarks. It defines instance classification as the pretext task [17]. Specifically, each image is considered as one class and is discriminated invariant to its own distortions and different from other images. [1] proposes to use the InfoNCE loss to recognize images by contrasting them with other images. [2] and [3] employ Siamese networks to improve the performance. However, a large number of images as negative samples are needed to contrast the positive sample. It requires a large batch size or a memory bank. Clustering-based approaches [18], [19] as the variants of contrastive learning keep consistency between cluster assignments for different views of images. Contrastive learning methods naturally align the embeddings of positive samples and separate the embeddings of negative samples to achieve uniformity of representation [20]. Our MVEB can be simplified to maximizing the alignment between view embeddings and the differential entropy of the embedding distribution. Maximizing the differential entropy can also improve uniformity without the need for a large batch size or memory bank.

Recent works can produce high-quality representation without negative samples in contrastive learning. Asymmetric network methods can learn representation by maximizing the agreement between the view embeddings based on Siamese networks. They rely on asymmetric network architecture to prevent collapse. BYOL [6] introduces a predictor network for the online branch and a momentum encoder. SimSiam [7] adopts a stop-gradient operation for the target branch and a predictor network to avoid collapse. DINO [8] combines the momentum encoder with self-distillation on the Transformer backbone. However, these methods are not well understood, and hard to interpret their architectural tricks [10]. Feature decorrelation methods [9], [10], [21], [22] avoid collapse by reducing the redundancy among the feature dimensions.

From the multi-view perspective, [13] proposes an information-theoretical framework to understand and explain the success of SSL. Specifically, two views of an image can be considered as the input and the self-supervised signals for each other. [12] proposes the multi-view information bottleneck for unsupervised multi-view learning to learn the minimal sufficient representation. However, the computation of mutual information is notoriously intractable. Our

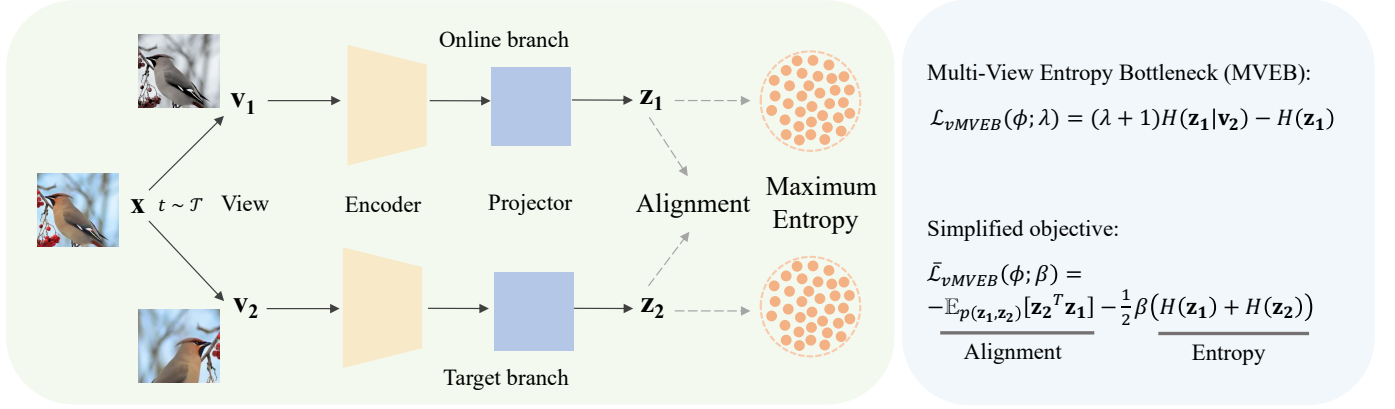


Fig. 2. Framework of MVEB and its training objective.

MVEB is related to the multi-view information bottleneck and can be effectively applied to Siamese networks to learn the minimal sufficient representation. [15] considers that the minimal sufficient representation gives rise to the degradation of performance for downstream tasks since not all task-relevant information is shared between views. Since either view is assumed (approximately) sufficient for downstream tasks for general data augmentation used in self-supervised learning [13], we argue that the non-shared task-relevant information between views can be ignored. Our experiment results also verify that the minimal sufficient representation can improve the generalization for downstream tasks.

3 PRELIMINARY: MINIMAL SUFFICIENT REPRESENTATION

Representation learning aims to transform input data $\mathbf{x}$ to the lower-dimensional representation $\mathbf{z}$ that contains the information relevant to the prediction task $\mathbf{y}$. This information is considered unchanged after encoding the data, which suggests $I(\mathbf{x}; \mathbf{y}) = I(\mathbf{z}; \mathbf{y})$, where I denotes mutual information. Thus, the learned representation $\mathbf{z}$ is sufficient for the prediction task [23].

Since $\mathbf{x}$ has more information than $\mathbf{y}$, the sufficient representation $\mathbf{z}$ of $\mathbf{x}$ may contain superfluous information irrelevant to the prediction task. The superfluous information can be represented as conditional mutual information $I(\mathbf{x}; \mathbf{z}|\mathbf{y})$. Among all sufficient representations, the minimal sufficient representation contains the least superfluous information.

4 APPROACH

We outline the general setting of training an encoder and a projector to learn a representation in the multi-view self-supervised setting in Fig. 2. The Siamese network consists of online and target branches. Each branch includes an encoder and a projector. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ be two different views of the input sample $\mathbf{x}$. We can get the ℓ_2 -normalized representations $\mathbf{z}_1$ and $\mathbf{z}_2$ from $\mathbf{v}_1$ and $\mathbf{v}_2$ through the deterministic function f_ϕ with the parameters ϕ . Let $q_\phi(\mathbf{z}_1)$ and $q_\phi(\mathbf{z}_2)$ be the marginal distributions of the representations $\mathbf{z}_1$ and $\mathbf{z}_2$, respectively, which are used to compute the entropy $H(\mathbf{z}_1)$ and $H(\mathbf{z}_2)$. We derive a new objective $\mathcal{L}_{vMVEB}(\phi; \lambda)$ to optimize the parameters ϕ .

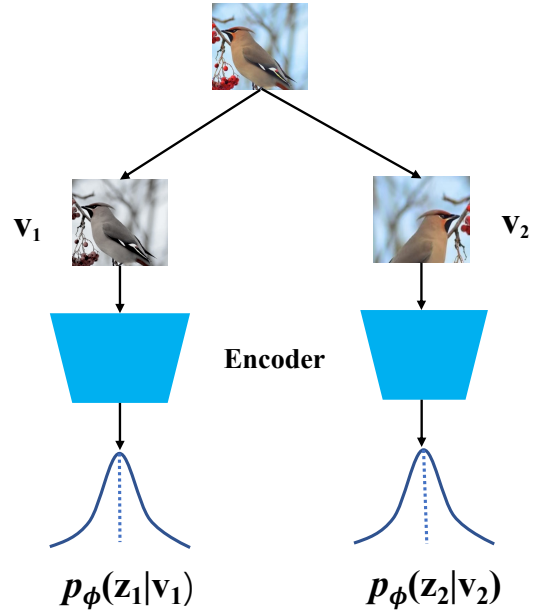


Fig. 3. Visualization of the multi-view information bottleneck model.

4.1 Multi-View Information Bottleneck

In the unsupervised setting, it is more challenging to obtain the minimal sufficient representation since the superfluous information cannot be identified without downstream tasks. To overcome the problem, [12] extends the information bottleneck theory in supervised learning to the multi-view unsupervised setting, termed multi-view information bottleneck (MVIB). The main idea relies on the multi-view assumption that either view is (approximately) sufficient for the prediction of downstream tasks and contains the same task-relevant information. In other words, the different views of a sample should not affect the prediction for downstream tasks. Similar to the role of labels in supervised learning, two views can be considered as mutual self-supervised signals for each other. Hence, we can obtain the sufficiency for downstream tasks by ensuring that the representation $\mathbf{z}_1$ of $\mathbf{v}_1$ is sufficient for $\mathbf{v}_2$. The superfluous information can also be identified as conditional mutual information $I_\phi(\mathbf{z}_1; \mathbf{v}_1|\mathbf{v}_2)$. Decreasing $I_\phi(\mathbf{z}_1; \mathbf{v}_1|\mathbf{v}_2)$ can achieve the

elimination of the superfluous information.

We can learn the minimal sufficient representation by satisfying these requirements using the relaxed Lagrangian multiplier method:

$$\mathcal{L}_{MVIB}(\phi; \lambda) = \lambda I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) - I_\phi(\mathbf{z}_1; \mathbf{v}_2). \quad (1)$$

Although MVIB is appealing in learning the minimal sufficient representation, the computation of the mutual information (MI) is notoriously intractable. To overcome the intractability in MVIB, [12] adopts the same stochastic network as VAE to obtain the Gaussian representation distributions $p(\mathbf{z}_1 | \mathbf{v}_1)$ and $p(\mathbf{z}_2 | \mathbf{v}_2)$ to approximately optimize $\mathcal{L}_{MVIB}$ as shown in Fig. 3. Hence, we obtain the variational MVIB objective as follows:

$$\begin{aligned} \mathcal{L}_{vMVIB}(\phi; \lambda) = & \lambda D_{KL}(p_\phi(\mathbf{z}_1 | \mathbf{v}_1) || p_\phi(\mathbf{z}_2 | \mathbf{v}_2)) \\ & - I_\phi(\mathbf{z}_1; \mathbf{z}_2), \end{aligned} \quad (2)$$

where D_{KL} denotes the Kullback-Leibler divergence. Specifically, $D_{KL}(p_\phi(\mathbf{z}_1 | \mathbf{v}_1) || p_\phi(\mathbf{z}_2 | \mathbf{v}_2))$ is the upper bound of $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$; $I_\phi(\mathbf{z}_1; \mathbf{z}_2)$ is the lower bound of $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ and approximated by InfoNCE [24], MINE [25] or MIGE [26].

4.2 Multi-View Entropy Bottleneck

MVIB cannot be directly applied to the Siamese network due to the intractability of the computation of mutual information. As mentioned in Section 4.1, the variational method can be introduced to optimize MVIB. However, this approximation optimization requires additional stochastic networks and does not work as expected in practice for visual recognition models compared with SimCLR, as shown in [15]. For the Siamese network, learning the minimal sufficient representation effectively in self-supervised representation learning is still challenging.

We derive the novel MVEB framework to solve the challenge of learning the minimal sufficient representation. Compared with MVIB, MVEB can be directly applied to the Siamese network without modification of the network structure and other complex designs.

The superfluous information $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ can be decomposed (see Appendix) as:

$$I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) = H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2), \quad (3)$$

where the conditional entropy $H(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2)$ contains no randomness (no information) as $\mathbf{z}_1$ being deterministic conditioned on $\mathbf{v}_1$. Hence, minimizing $H(\mathbf{z}_1 | \mathbf{v}_2)$ is equivalent to minimizing $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$. We can also decompose $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ (see Appendix) as:

$$I_\phi(\mathbf{z}_1; \mathbf{v}_2) = H(\mathbf{z}_1) - H(\mathbf{z}_1 | \mathbf{v}_2). \quad (4)$$

Based on the above derivations and Eq. (1), we obtain the general MVEB objective:

$$\mathcal{L}_{MVEB}(\phi; \lambda) = (\lambda + 1)H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1), \quad (5)$$

where $H(\mathbf{z}_1 | \mathbf{v}_2) = -\mathbb{E}_{p(\mathbf{z}_1 | \mathbf{v}_2)}[\log p(\mathbf{z}_1 | \mathbf{v}_2)]$. We can learn the minimal sufficient representation with $\mathcal{L}_{MVEB}$ for the deterministic encoder.

The conditional entropy $H(\mathbf{z}_1 | \mathbf{v}_2)$ is intractable since the distribution $p(\mathbf{z}_1 | \mathbf{v}_2)$ is unknown. To overcome this problem, we introduce $q_\phi(\mathbf{z}_1 | \mathbf{v}_2)$ that is a variational approximation to $p(\mathbf{z}_1 | \mathbf{v}_2)$. Since $KL(p(\mathbf{z}_1 | \mathbf{v}_2) || q_\phi(\mathbf{z}_1 | \mathbf{v}_2)) \geq 0$, we can derive the upper bound of $H(\mathbf{z}_1 | \mathbf{v}_2)$:

$$\begin{aligned} H(\mathbf{z}_1 | \mathbf{v}_2) &= -\mathbb{E}_{p(\mathbf{z}_1 | \mathbf{v}_2)}[\log p(\mathbf{z}_1 | \mathbf{v}_2)] \\ &\leq -\mathbb{E}_{p(\mathbf{z}_1 | \mathbf{v}_2)}[\log q_\phi(\mathbf{z}_1 | \mathbf{v}_2)]. \end{aligned} \quad (6)$$

Hence, a variational MVEB term (vMVEB) is defined as

$$\begin{aligned} \mathcal{L}_{vMVEB}(\phi; \lambda) &= -(\lambda + 1)\mathbb{E}_{p(\mathbf{z}_1 | \mathbf{v}_2)}[\log q_\phi(\mathbf{z}_1 | \mathbf{v}_2)] \\ &\quad - H(\mathbf{z}_1). \end{aligned} \quad (7)$$

For self-supervised learning based on the Siamese network, e.g. SimCLR, BYOL, and SimSiam, the representation $\mathbf{z}_1$ is ℓ_2 -normalized in the hypersphere space to improve the performance of the models. We also normalize $\mathbf{z}_1$ in the hypersphere space. The von Mises-Fisher (vMF) is the common distribution of the hypersphere space. Hence, we define $q_\phi(\mathbf{z}_1 | \mathbf{v}_2)$ as the vMF distribution, i.e.,

$$q_\phi(\mathbf{z}_1 | \mathbf{v}_2) = C_n(\kappa) e^{\kappa \boldsymbol{\mu}^T \mathbf{z}_1}, \quad (8)$$

where $\boldsymbol{\mu}$ is the mean direction, κ denotes the concentration parameter of the von Mises-Fisher distribution and $C_n(\kappa) = \frac{\kappa^{\frac{n}{2}-1}}{(2\pi)^{n/2} I_{\frac{n}{2}-1}(\kappa)}$ is the normalization function of κ . We assume that κ is a constant and $q_\phi(\mathbf{z}_1 | \mathbf{v}_2)$ is parameterized by $\boldsymbol{\mu}$. As illustrated in Fig. 2, we use the target branch to encode $\mathbf{v}_2$ and output $\mathbf{z}_2$ as $\boldsymbol{\mu}$. Hence, we further obtain

$$H(\mathbf{z}_1 | \mathbf{v}_2) \leq -\log C_n(\kappa) - \kappa \mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1], \quad (9)$$

which allows us to reformulate the objective of Eq. (7) as follows,

$$\hat{\mathcal{L}}_{vMVEB}(\phi; \beta) = -\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1] - \beta H(\mathbf{z}_1), \quad (10)$$

where $\beta = \frac{1}{(\lambda+1)\kappa}$ is the balance factor. This simplified objective maximizes both the agreement between $\mathbf{z}_1$ and $\mathbf{z}_2$ and the differential entropy of $\mathbf{z}_1$ to learn the minimal sufficient representation for the deterministic encoder.

The sample view $\mathbf{v}_1$ can also be regarded as the self-supervised signal for $\mathbf{v}_2$. Similarly, we derive another optimization objective. The final simplified training objective is obtained as follows,

$$\begin{aligned} \bar{\mathcal{L}}_{vMVEB}(\phi; \beta) &= -\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1] \\ &\quad - \frac{1}{2}\beta(H(\mathbf{z}_1) + H(\mathbf{z}_2)). \end{aligned} \quad (11)$$

However, it is unfortunately intractable to compute $H(\mathbf{z}_1)$ and $H(\mathbf{z}_2)$ since the distributions of $\mathbf{z}_1$ and $\mathbf{z}_2$ are unknown. We propose a score-based entropy estimator with the von Mises-Fisher kernel to maximize $H(\mathbf{z}_1)$ and $H(\mathbf{z}_2)$, which is described in Section 4.4. The training pseudocode is given in Algorithm 1.

4.3 Analysis of the Variational Approximation

In our work, the variational approximation is used to obtain the upper bound of the conditional entropy $H(\mathbf{z}_1 | \mathbf{v}_2)$ for minimization, rather than estimate the true $H(\mathbf{z}_1 | \mathbf{v}_2)$. However, we need to guarantee that this upper bound is not loose to achieve the minimization of $H(\mathbf{z}_1 | \mathbf{v}_2)$.

The bound of the variational approximation of $H(\mathbf{z}_1|\mathbf{v}_2)$ in Eq. (6) is completely tight if $\mathbb{E}_{p(\mathbf{v}_2)}[KL(p(\mathbf{z}_1|\mathbf{v}_2)||q_\phi(\mathbf{z}_1|\mathbf{v}_2))] = 0$. It means that $q_\phi(\mathbf{z}_1|\mathbf{v}_2)$ equals $p(\mathbf{z}_1|\mathbf{v}_2)$. In other words, if $\mathbb{E}_{p(\mathbf{v}_2)}[KL(p(\mathbf{z}_1|\mathbf{v}_2)||q_\phi(\mathbf{z}_1|\mathbf{v}_2))]$ is smaller, the bound of the variational approximation is tighter. Below we prove the bound of the variational approximation is not loose and can achieve the minimization of $H(\mathbf{z}_1|\mathbf{v}_2)$.

If the approximation is very loose, we cannot optimize $H(\mathbf{z}_1|\mathbf{v}_2) = -\mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log p(\mathbf{z}_1|\mathbf{v}_2)]$; in other words, $-\mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log p(\mathbf{z}_1|\mathbf{v}_2)]$ cannot be reduced during optimization. Decompose $\mathbb{E}_{p(\mathbf{v}_2)}[KL(p(\mathbf{z}_1|\mathbf{v}_2)||q_\phi(\mathbf{z}_1|\mathbf{v}_2))]$ as follows:

$$\begin{aligned} & \mathbb{E}_{p(\mathbf{v}_2)}[KL(p(\mathbf{z}_1|\mathbf{v}_2)||q_\phi(\mathbf{z}_1|\mathbf{v}_2))] \\ &= \mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log p(\mathbf{z}_1|\mathbf{v}_2)] - \mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log q_\phi(\mathbf{z}_1|\mathbf{v}_2)]. \end{aligned} \quad (12)$$

Since the first term $\mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log p(\mathbf{z}_1|\mathbf{v}_2)]$ of Eq. (12) is fixed, minimizing $-\mathbb{E}_{p(\mathbf{z}_1, \mathbf{v}_2)}[\log q_\phi(\mathbf{z}_1|\mathbf{v}_2)]$ is equivalent to minimizing $\mathbb{E}_{p(\mathbf{v}_2)}[KL(p(\mathbf{z}_1|\mathbf{v}_2)||q_\phi(\mathbf{z}_1|\mathbf{v}_2))]$, which makes the bound tight.

Algorithm 1 MVEB pytorch pseudocode.

```
# f: encoder containing a backbone and
# a projector
# lambda: loss balance coefficient
# N: batch size

for x in loader: # load a minibatch x

    # augmentation
    v_1, v_2 = augment(x)

    # compute normalized embeddings
    z_1, z_2 = f(v_1), f(v_2)

    # Alignment loss
    Align_loss = mm(z_1, z_2.t()).mean()

    # compute the score function S(.)
    # SGE: Stein Gradient Estimator
    S(z_1) = SGE(z_1)
    S(z_2) = SGE(z_2)

    # compute the entropy loss
    En_z_1 = (S(z_1).detach() * z_1).sum(-1).mean()
    En_z_2 = (S(z_2).detach() * z_2).sum(-1).mean()

    # compute the total loss
    loss = Align_loss + 0.5 * lambda * (En_z_1 + En_z_2)

    # optimization step
    loss.backward()
    optimizer.step()
```

4.4 Score-Based Entropy Estimation with the von Mises-Fisher Kernel

For learning the minimal sufficient representation with MVEB, we need to maximize $H(\mathbf{z}) = -\mathbb{E}_{q_\phi(\mathbf{z})}[\log q_\phi(\mathbf{z})]$. We first analyze the gradient of $H(\mathbf{z})$ w.r.t. ϕ , which can be decomposed as:

$$\nabla_\phi H(\mathbf{z}) = -\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[\log q(\mathbf{z})] - \mathbb{E}_{q(\mathbf{z})}[\nabla_\phi \log q_\phi(\mathbf{z})], \quad (13)$$

where $q(\mathbf{z})$ without the subscript ϕ means the gradient of computation is irrelevant to ϕ . The second term on the right part of Eq. (13) can be further decomposed as:

$$\begin{aligned} \mathbb{E}_{q(\mathbf{z})}[\nabla_\phi \log q_\phi(\mathbf{z})] &= \mathbb{E}_{q(\mathbf{z})}[\nabla_\phi q_\phi(\mathbf{z}) \times \frac{1}{q(\mathbf{z})}] \\ &= \int \nabla_\phi q_\phi(\mathbf{z}) d\mathbf{z} = \nabla_\phi \int q_\phi(\mathbf{z}) d\mathbf{z} = 0. \end{aligned} \quad (14)$$

Hence we have

$$\nabla_\phi H(\mathbf{z}) = -\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[\log q(\mathbf{z})]. \quad (15)$$

However, $\nabla_\phi H(\mathbf{z})$ is non-trivial because the expectation w.r.t. $q_\phi(\mathbf{z})$ is not differentiable w.r.t. ϕ .

To overcome this problem, we adopt the general reparameterization trick proposed in [27] for the computation of $\nabla_\phi H(\mathbf{z})$. In detail, the samples from the representation distribution $q_\phi(\mathbf{z})$ can be obtained by encoding the data samples $\mathbf{v}, \mathbf{z} = f_\phi(\mathbf{v})$, where f_ϕ is the deterministic function (encoder and projector). Hence, the representation can be reparameterized via the following differentiable transformation:

$$\mathbf{z} = f_\phi(\mathbf{v}) \quad \text{with} \quad \mathbf{v} \sim p(\mathbf{v}). \quad (16)$$

Since $p(\mathbf{v})$ is irrelevant to the model parameters ϕ , the expectation w.r.t. $q_\phi(\mathbf{z})$ can be rewritten via the above reparameterization, which makes the expectation differentiable w.r.t. ϕ . Hence, the entropy gradient estimator is derived as follows:

$$\begin{aligned} \nabla_\phi H(\mathbf{z}) &= -\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{z})}[\log q(\mathbf{z})] = -\mathbb{E}_{p(\mathbf{v})}[\nabla_\phi \log q(f_\phi(\mathbf{v}))] \\ &= -\mathbb{E}_{p(\mathbf{v})}[\nabla_{\mathbf{z}} \log q(\mathbf{z}) \nabla_\phi f_\phi(\mathbf{v})], \end{aligned} \quad (17)$$

where $\nabla_{\mathbf{z}} \log q(\mathbf{z})$ is the score function, which can be directly approximated by score estimation using a black-box function [28]. $\nabla_\phi f_\phi(\mathbf{x})$ can be obtained by direct back-propagation. As long as we can provide a good enough approximation to the score function, this estimation of the entropy gradient is approximately unbiased.

The Stein gradient estimator described in Appendix is an effective estimation of the score function [28]. We adopt it to approximate the score function $S(\mathbf{z}) = \nabla_{\mathbf{z}} \log q(\mathbf{z})$. Since the representation $\mathbf{z}$ is ℓ_2 -normalized, we propose to use the following von Mises-Fisher kernel to compute $S(\mathbf{z})$:

$$k(\mathbf{z}, \mathbf{z}') = \exp\left(\frac{\mathbf{z}^T \mathbf{z}'}{\Delta}\right), \quad (18)$$

where Δ is the bandwidth of the von Mises-Fisher kernel. We set it to the median of pairwise cosine distances among all samples in the batch.

5 RETHINKING ALIGNMENT AND UNIFORMITY

Contrastive learning (e.g., SimCLR and MOCO) aims to bring similar (positive) samples closer and dissimilar (negative) samples farther apart. [20] decomposes the contrastive loss into alignment and uniformity. As shown in [29], asymmetric network methods (e.g., BYOL and SimSiam) and feature decorrelation methods (e.g., Barlow Twins and VICReg) are viewed as optimizing alignment and uniformity based on gradient analysis. Asymmetric network methods rely on a predictor to optimize the uniformity, and feature decorrelation methods rely on feature decorrelation to optimize the uniformity.

We consider the multi-view self-supervised setting in Fig. 2, where $\mathbf{z}_1$ and $\mathbf{z}_2$ are the representations of views $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. If we consider $\mathbf{v}_2$ as the supervised information, minimizing superfluous information $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ is equivalent to minimizing $H(\mathbf{z}_1 | \mathbf{v}_2)$ for the deterministic encoder, as shown in Section 4.2. Furthermore, maximizing the alignment $\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1]$ is equivalent to minimizing $H(\mathbf{z}_1 | \mathbf{v}_2)$ (see Eq. (9)). Hence, maximizing the alignment can eliminate superfluous information.

We find that maximizing the mutual information $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ between $\mathbf{z}_1$ and the supervised information $\mathbf{v}_2$ not only keeps the information relevant to $\mathbf{v}_2$ but also reduces superfluous information. This is because $I_\phi(\mathbf{z}_1; \mathbf{v}_2) = H(\mathbf{z}_1) - H(\mathbf{z}_1 | \mathbf{v}_2)$ and minimizing superfluous information $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ is equivalent to minimizing $H(\mathbf{z}_1 | \mathbf{v}_2)$ for the deterministic encoder. As uniformity prefers the feature distribution $p(\mathbf{z}_1)$ that preserves its maximal entropy $H(\mathbf{z}_1)$, we can consider $H(\mathbf{z}_1)$ as the uniformity. Hence, $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ is the combination of alignment and uniformity. In another view, maximizing alignment and uniformity can keep the information in $\mathbf{z}_1$ relevant to $\mathbf{v}_2$. However, since the superfluous information is not minimal, maximizing $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ cannot achieve the goal of learning of minimal sufficient representation.

In this work, we present a new objective function, $\mathcal{L}_{MVEB}(\phi; \lambda) = (\lambda + 1)H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1)$, to learn the minimal sufficient representation. $H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1) = -I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ aims to keep the information relevant to $\mathbf{v}_2$, and $\lambda H(\mathbf{z}_1 | \mathbf{v}_2)$ aims to reduce superfluous information $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$. Since maximizing the alignment $\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1]$ is equivalent to minimizing $H(\mathbf{z}_1 | \mathbf{v}_2)$, the balance of maximizing alignment and uniformity can learn minimal sufficient representation. Specifically, the coefficient β in Eq. (11) is used to balance the optimization of the alignment and $H(\mathbf{z}_1) + H(\mathbf{z}_1)$. In Eq. (10), $\beta = \frac{1}{(\lambda+1)\kappa}$, where κ is a constant. As shown in Eq. (1), λ is the coefficient to balance the optimization of $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ and $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$. Increasing β to make λ smaller than a threshold does not eliminate superfluous information effectively, which hurts the performance of downstream tasks. Decreasing β to approach zero means λ approaches infinity, and the Siamese network suffers from model collapse with the trivial constant representations without maximizing the uniformity. In another view, when λ approaches infinity, the optimization objective in Eq. (1) only considers minimizing superfluous information $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$. Hence, a trivial solution is obtained if constant representations of $\mathbf{v}_1$ are outputted, which means the representations do not contain the information

relevant to $\mathbf{v}_1$ and $\mathbf{v}_2$.

Relation with Contrastive Learning. The contrastive loss, also termed InfoNCE, is a lower bound of $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ [13]. More negative samples make this lower bound tighter. Maximizing the contrastive loss aims to maximize $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$. Specifically, when the number of negative samples $N \rightarrow \infty$, the normalized contrastive loss reaches the following convergence:

$$\begin{aligned} \lim_{N \rightarrow \infty} \mathcal{L}_{\text{contrastive}}(\phi; \tau, N) - \log N \\ = -\frac{1}{\tau} \mathbb{E}_{(\mathbf{v}_1, \mathbf{v}_2) \sim p_{\text{pos}}} \left[f_\phi(\mathbf{v}_1)^\top f_\phi(\mathbf{v}_2) \right] \\ + \mathbb{E}_{\mathbf{v}_1 \sim p_{\text{data}}} \left[\log \mathbb{E}_{\mathbf{v}_1^- \sim p_{\text{data}} \setminus \mathbf{v}_1} \left[e^{f_\phi(\mathbf{v}_1^-)^\top f_\phi(\mathbf{v}_1) / \tau} \right] \right], \quad (19) \end{aligned}$$

where $\mathbf{v}_1^-$ denotes the negative samples of $\mathbf{v}_1$ and p_{pos} denotes the distribution of the pairs of positive samples. The first term of the right-hand side of Eq. (19) aims to maximize alignment. The second term pushes dissimilar (negative) samples farther apart. The performances of contrastive learning are sensitive to the choice of the hyperparameter τ , since τ is used to balance the optimization of alignment and uniformity to learn minimal sufficient representation. However, τ is also incorporated to maximize uniformity, which limits the balance between alignment and uniformity based on the gradient analysis in [29].

Contrastive learning relies on a large number of negative samples to optimize uniformity and keep a lower bound of $I(\mathbf{z}_1; \mathbf{v}_2)$ tight. As uniformity optimization is based on the separation of the instances, it is hard to maximize global uniformity effectively. Unlike contrastive learning, our MVEB directly maximizes the differential entropy of the global feature distribution, which is, in principle, more effective for uniformity maximization.

Relation with Asymmetric Network Methods and Feature Decorrelation Methods. As shown in [29], from the gradient analysis, asymmetric network methods and feature decorrelation methods can be unified into the same form:

$$\mathcal{L}(\phi; \lambda) = -\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1] + \lambda \mathbb{E}_{p(\mathbf{z}_1)}[\mathbf{z}_1^T F \mathbf{z}_1], \quad (20)$$

where F is the correlation matrix of features; λ is the balance factor. For asymmetric network methods, F is updated according to the moving average; for feature decorrelation methods, F is computed according to the features of each batch. The first term of the right-hand side of Eq. (20) is the alignment. The second term is the derivation of the following entry:

$$\begin{aligned} \mathbb{E}_{p(\mathbf{z}_1^-)}[\cos^2(\mathbf{z}_1, \mathbf{z}_1^-)] &= \mathbb{E}_{p(\mathbf{z}_1^-)}[\mathbf{z}_1^T \mathbf{z}_1^- \mathbf{z}_1^{-T} \mathbf{z}_1] \\ &= \mathbb{E}_{p(\mathbf{z}_1^-)}[\mathbf{z}_1^T (\mathbf{z}_1^- \mathbf{z}_1^{-T}) \mathbf{z}_1] = \mathbf{z}_1^T F \mathbf{z}_1, \quad (21) \end{aligned}$$

where $\mathbf{z}_1^-$ denotes the negative samples of $\mathbf{z}_1$. According to this derivation, the second term of Eq. (20) aims to minimize the similarity between negative samples to maximize uniformity. Hence, asymmetric network methods and feature decorrelation methods also achieve balancing the optimization of alignment and uniformity to learn minimal sufficient representation.

Asymmetric network methods rely on a predictor to optimize uniformity; feature decorrelation methods rely on

feature decorrelation to optimize uniformity. This difference leads to the difference in performances. Although the performances of these methods are better than contrastive learning, they still rely on separating the instances, which poses difficulty in achieving optimal results. We argue that maximizing the differential entropy is the better way to learn the minimal sufficient representation with the feature distribution rather than the instance separation.

6 MAIN RESULTS

We first assess MVEB’s representation with the self-supervised benchmark on ImageNet dataset [30] in linear evaluation. We then evaluate our model by transferring it to other datasets and tasks, including image classification, object detection, and segmentation.

6.1 Pretraining Details

We pretrain our model MVEB on ImageNet with ResNet-50 as the backbone. The projector network consists of three linear layers, each with an output dimensionality set to 2048. We apply BN and ReLU after the first two layers. According to our empirical study, the momentum encoder is chosen as the target branch (see Fig. 2). Following the setting of BYOL [6], we update the target branch with the exponential moving by increasing the average parameters from 0.996 to 1 with a cosine scheduler.

We follow the strategy of image augmentation used in BYOL, including random cropping, color jittering, converting to grayscale, horizontal flipping, Gaussian blurring and solarization. We also adopt multi-crop to get six local views [8] of 96×96 . All augmentation parameters are the same as those in DINO [8]. The local views are only passed through the online branch. In addition, the positive sample of each local view only comes from the average of the embeddings of the two global views [8] from the same sample.

We train MVEB for 800 epochs with the LARS [31] optimizer. The weight decay and the momentum are set to $1e-6$ and 0.9, respectively. The basic learning rate is 0.4, scaled with the batch size and divided by 256. We decrease the learning rate to one-thousandth with a cosine decay scheduler after a warm-up period of 10 epochs. The biases and batch normalization parameters are excluded from the LARS adaptation. The batch size is 4096, distributed over 32 NVIDIA V100 GPUs. The coefficient β in the loss function is set to 0.01 according to our empirical study.

6.2 Linear Evaluation on ImageNet

Following the ImageNet linear evaluation in [2], [6], [38] and [3], we train a linear classifier on top of the frozen learned representation to assess the classification performance on ImageNet [30]. The number of training epochs is set to 50. Other training settings of the linear evaluation are kept the same as [7].

We compare MVEB with previous popular SSL methods based on the Siamese network. The results are shown in Table 1. MVEB significantly exceeds the previous best method UniGrad in top-1 accuracy by an absolute improvement of 1.4%. Compared with the supervised baseline used in [2],

MVEB surpasses the supervised result of 76.5%. To the best of our knowledge, MVEB is the first work that exceeds this supervised learning result with the vanilla ResNet-50 backbone.

We compare MVEB with the masked autoencoder-based methods, MAE [36] and SimMM [37]. Table 2 shows the result. For linear evaluation on ImageNet, MVEB outperforms MAE and SimMM. Moreover, the number of parameters of MVEB is the smallest.

6.3 Semi-Supervised Classification on ImageNet

We implement the semi-supervised learning by fine-tuning the pre-trained MVEB on both the 1% and 10% subsets of the ImageNet training set, utilizing the same partitions as in SimCLR. Adhering to the semi-supervised training configurations outlined in [10], we train a linear classifier and fine-tune the representations using 1% and 10% of the available labels. Our training employs the SGD optimizer with no weight decay, a batch size of 256, and running for 60 epochs. For the training with 1% of labels, we use a learning rate of 0.002 for the encoder and a learning rate of 0.8 for the linear head. For the training with 10% of labels, we use a learning rate of 0.003 for the encoder and a learning rate of 0.4 for the linear head. The cosine decay is employed to adjust the two learning rates. The augmentation pipelines used for training data and validation are the same as those for augmenting the data in linear evaluation.

In Table 3, we present the top-1 and top-5 accuracies. Our results indicate that MVEB outperforms previous methods consistently in both the 1% and 10% settings. Additionally, it is worth mentioning that all self-supervised learning methods significantly outperform the supervised baseline [39].

6.4 Transfer Learning

To assess whether our learned representation is generic across different domains, we transfer it to other classification tasks on 11 datasets, including FGVC-Aircraft [40], Caltech-101 [41], Stanford Cars [42], CIFAR-10 [43], CIFAR-100 [43], Describable Textures Dataset (DTD) [44], Oxford 102 Flowers [45], Food-101 [46], Oxford-IIIT Pets [47], SUN397 [48] and Pascal VOC2007 [49]. For each dataset, we conduct (a) linear evaluation, where a multinomial logistic regression model is fit on top of the embeddings extracted from the frozen ResNet-50 backbone, and (b) fine-tuning, where the weights of both the backbone and classifier are allowed to be updated. We search for the best hyperparameters (ℓ_2 -regularization coefficient for linear evaluation and learning rate and weight decay for fine-tuning) on the split validation set and report the evaluation result on the test set of each dataset.

Following the common practice in [2] and [6], we evaluate the transfer performance on the 11 datasets via linear classification and fine-tuning. As for the evaluation, we use the metrics in the papers introducing these datasets. Specifically, we report top-1 accuracy for CIFAR-10, CIFAR-100, DTD, Food-101, Stanford Cars and SUN397, mean per-class accuracy for Caltech-101, FGVC-Aircraft, Oxford-IIIT Pets, and Oxford 102 Flowers, and 11-point mAP from [49] for Pascal VOC 2007. For DTD and SUN397, which contain

TABLE 1

Top-1 and top-5 accuracies (%) of linear classification on ImageNet [30]. The bold entries denote the best. The results of all methods are based on the ResNet-50 [32] backbone for a fair comparison.

Method	Batch size	Training Epoch	Multi-crop	Top-1	Top-5
SimCLR [2]	4096	1000	2×224	69.3	89.0
MoCo-v2 [33]	256	800	2×224	71.1	90.1
InfoMin [34]	256	800	2×224	73.0	91.1
SimSiam [7]	256	800	2×224	71.3	-
Barlow Twins [9]	2048	1000	2×224	73.2	91.0
VICReg [10]	2048	1000	2×224	73.2	91.1
BYOL [6]	4096	1000	2×224	74.3	91.6
MVEB (ours)	4096	800	2×224	74.6	92.1
SwAV [19]	4096	800	$2 \times 224 + 6 \times 96$	75.3	-
Self-Classifer [35]	4096	800	$2 \times 224 + 6 \times 96$	74.1	-
DINO [8]	4096	800	$2 \times 224 + 8 \times 96$	75.3	92.5
UniGrad [29]	4096	800	$2 \times 224 + 6 \times 96$	75.5	-
MVEB (ours)	4096	800	$2 \times 224 + 6 \times 96$	76.9	93.3

TABLE 2

Comparison of MVEB with masked autoencoder-based methods for linear evaluation on ImageNet [30]. The bold entry denotes the best.

Method	Batch size	Epoch	Backbone	Param.	Top-1
MAE [36]	4096	1600	ViT-B	85M	67.8
MAE [36]	4096	1600	ViT-L	307M	76.0
SimMM [37]	2048	800	ViT-B	85M	56.7
MVEB (ours)	4096	800	ResNet-18	11.7M	64.9
MVEB (ours)	4096	800	ResNet-34	21.8M	69.6
MVEB (ours)	4096	800	ResNet-50	23M	76.9
MVEB (ours)	4096	800	ResNet-101	44.5M	78.2

TABLE 3

Semi-supervised classification results on ImageNet. We use 1% and 10% training examples to fine-tune the pre-trained model.

Method	1%		10%	
	Top 1	Top 5	Top 1	Top 5
Supervised [39]	25.4	48.4	56.4	80.4
SimCLR [2]	48.3	75.5	65.6	87.8
BYOL [6]	53.2	78.4	68.8	89.0
Barlow Twins [9]	55.0	79.2	69.7	89.3
DINO [8]	52.2	78.2	68.2	89.1
VICReg [10]	54.8	79.4	69.5	89.5
MVEB (ours)	57.5	82.1	72.6	91.5

multiple train/test splits defined by the original creators, we only report results in the first train/test split. For Caltech-101, since there is no defined train/test split, we randomly select 30 images per class to form the training set and the rest is for test. DTD, FGVC-Aircraft, Pascal VOC 2007, and Oxford 102 Flowers have their own validation sets, and we directly use them. For the others, we hold out a subset by randomly selecting 20% from the training set to form the validation set. The hyperparameters are chosen based on the metrics on the split validation set, and the final results are reported on the test set.

Linear Classification. We fit an ℓ_2 -regularization multinomial logistic regression model on top of the embeddings extracted from the frozen ResNet-50 backbone. The images are resized to 224 pixels along the shorter side using bicubic

resampling, after which a center crop of 224×224 is followed. We use L-BFGS [50] to optimize the softmax cross-entropy objective. The coefficient of the ℓ_2 -regularization for each dataset is chosen on the validation set, ranging over a grid of 45 logarithmically spaced values between 10^{-6} and 10^5 .

Fine-Tuning. We initialize the model with the parameters of the pretrained model and tune the whole network. For augmentation, we only perform random crops with resizing and flipping at training time. With a batch size of 64, we train the model for 5000 iterations. The optimizer is SGD with the Nesterov momentum with the parameter set to 0.9. The learning rate is decreased with the cosine annealing schedule without restart. We search for the best learning rate and weight decay on the validation set. Specifically, the initial learning rate is chosen from a grid of 4 logarithmically spaced values between 0.0001 and 0.1, and the weight decay is chosen from a grid of 4 logarithmically spaced values between 10^{-6} and 10^{-3} , as well as no weight decay. The values of weight decay are divided by the learning rates.

As shown in Table 4, for linear evaluation, MVEB outperforms other methods by a large margin on all datasets except for DTD and Pets where the results of MVEB are still competitive. In the case of fine-tuning, MVEB also achieves the best or second best on 9 of 11 datasets, surpassing the supervised baseline in terms of the average evaluation metric of all datasets. Compared with other SSL methods, MVEB shows more advantages in generalization across different image domains.

6.5 Object Detection and Segmentation

We further evaluate the learned embeddings by transferring them to more downstream tasks besides classification, including object detection and instance segmentation on MS COCO [54]. We adopt ResNet-50 [32] with the feature pyramid network (FPN) [54] and Mask RCNN [55] for detection and segmentation. The ResNet-50 backbone is pretrained by MVEB for 800 epochs, as in Section 6.2. For implementation, we adopt Detectron2 [56] and use the hyperparameters suggested in [57] without searching for the

TABLE 4

Transfer learning results of the models pretrained on ImageNet. The best result for each dataset is **bold**, and the second best is underlined. ‘Supervised’ refers to the pretrained model provided in the PyTorch library [51], where the model is pretrained using the labels of ImageNet. The results of all methods are based on the vanilla ResNet-50 backbone.

Method	Aircraft	Caltech101	Cars	CIFAR10	CIFAR100	DTD	Flowers	Food	Pets	SUN397	VOC2007	Avg.
<u>Linear evaluation</u>												
Supervised	43.6	90.2	44.9	91.4	73.9	72.2	89.9	69.5	91.5	60.5	83.6	73.8
InfoMin [34]	38.6	87.8	41.0	91.5	73.4	74.7	87.2	69.5	86.2	61.0	83.2	72.2
SimCLR [2]	44.9	90.1	43.7	91.2	72.7	74.2	90.9	67.5	83.3	59.2	80.8	72.6
MoCo v2 [33]	41.8	87.9	39.3	92.3	74.9	73.9	90.1	69.0	83.3	60.3	82.7	72.3
SimCLR v2 [52]	46.4	89.6	50.4	92.5	76.8	76.4	92.9	<u>73.1</u>	84.7	61.5	81.6	75.1
BYOL [6]	<u>53.9</u>	<u>91.5</u>	<u>56.4</u>	<u>93.3</u>	<u>77.9</u>	76.9	<u>94.5</u>	73.0	89.1	60.0	81.1	<u>77.1</u>
MVEB (ours)	55.8	92.7	59.6	94.6	79.4	<u>76.5</u>	95.7	77.9	<u>91.2</u>	66.5	85.5	79.6
<u>Fine-tuning</u>												
Supervised	83.5	91.0	82.6	96.4	82.9	73.3	<u>95.5</u>	84.6	92.4	63.6	<u>84.8</u>	84.6
InfoMin [34]	80.2	83.9	78.8	96.9	71.2	71.1	95.2	78.9	85.3	57.7	76.6	79.6
SimCLR [2]	81.1	<u>90.4</u>	83.8	97.1	84.5	71.5	93.8	82.4	84.1	63.3	82.6	83.1
MoCo v2 [33]	79.9	84.4	75.2	96.5	71.3	69.5	94.4	76.8	79.8	55.8	71.7	77.7
SimCLR v2 [52]	78.7	82.9	79.8	96.2	79.1	70.2	94.3	82.2	83.2	61.1	78.2	80.5
BYOL [6]	79.5	89.4	<u>84.6</u>	<u>97.0</u>	<u>84.0</u>	<u>73.6</u>	94.5	85.5	89.6	<u>64.0</u>	82.7	84.0
MVEB (ours)	85.9	88.6	89.9	<u>97.0</u>	80.4	74.5	95.7	<u>84.9</u>	<u>91.9</u>	64.3	85.6	85.3

TABLE 5

Object detection and instance segmentation results (%) on MS COCO.

Method	COCO object detection			COCO instance segmentation		
	AP_{all}^{box}	AP_{50}^{box}	AP_{75}^{box}	AP_{all}^{mask}	AP_{50}^{mask}	AP_{75}^{mask}
Supervised	38.9	59.6	42.7	35.4	56.5	38.1
MoCo v2 [33]	39.8	59.8	43.6	36.1	56.9	38.7
DenseCL [53]	40.3	59.9	44.3	36.4	57.0	39.2
DC v2 [19]	41.0	61.8	45.1	37.3	58.7	39.9
DINO [8]	41.4	62.2	45.3	37.5	58.8	40.2
SimCLR [2]	41.6	61.8	45.6	37.6	59.0	40.5
UniGrad [29]	42.0	62.6	45.7	37.9	59.7	40.7
MVEB (ours)	42.2	62.8	46.2	38.1	59.8	41.1

best hyperparameters. The model is fine-tuned on COCO 2017 with the $1 \times$ training schedule [3].

The results are reported in Table 5. It is shown that our MVEB consistently outperforms other methods on both object detection and instance segmentation w.r.t. all evaluation metrics. This indicates that MVEB’s representation generalizes well beyond the ImageNet classification task.

7 EMPIRICAL STUDY

In this section, we explore the behaviors of MVEB in self-supervised learning with Siamese networks. In all empirical studies, our model based on ResNet-50 [32] backbone is pretrained for 100 epochs on ImageNet [30]. We report all results using the linear evaluation protocol on ImageNet [7]. A supervised linear classifier is trained based on the frozen features from the pre-trained model and the number of training epochs is set to 50. Other training settings of the linear evaluation are kept the same as [7].

7.1 Batch Size

Empirical experiments are conducted to evaluate the performance of our method with different batch sizes. We compare

MVEB with SimCLR [2], SimSiam [7] and VICReg [10]. We use a symmetric Siamese network without a predictor network, a momentum encoder, and a stop-gradient operation. The batch size is set to a range from 128 to 4096. The projector network consists of three linear layers, each with the output dimensionality set to 8194. We apply BN and ReLU after the first two layers. SGD is used as the optimizer. The weight decay and the momentum are set to $1e-4$ and 0.9, respectively. The basic learning rate is 0.05, scaled with the batch size and divided by 256, and the loss function coefficient β is set to 0.01.

The results are reported in Table 6. MVEB works well over a wide range of batch-size settings. We can observe that the top-1 accuracy of MVEB increases as the batch size increases. When the batch size varies from 512 to 4096, the accuracies of MVEB are similar. Compared with SimCLR, SimSiam, and VICReg, our MVEB outperforms them by a large margin with different batch sizes.

Although MVEB adopts the Siamese network with direct weight-sharing similar to SimCLR, MVEB can work well without the requirement of a large batch (e.g., 4096). The behavior of MVEB is also different from SimSiam and VICReg, both of which, with the large batch size 4096,

TABLE 6
Top-1 accuracies (%) of linear classification on ImageNet of SSL methods pretrained with different batch sizes.

Batch size	128	256	512	1024	2048	4096
SimCLR [2]	-	57.5	60.7	62.8	64.0	64.6
SimSiam [7]	67.3	68.1	68.1	68.0	67.9	64.0
VICReg [10]	67.3	67.9	68.2	68.3	68.6	67.8
MVEB (ours)	67.8	68.5	68.8	68.9	68.9	69.0

drop significantly in accuracy. Moreover, SimSiam relies on the predictor network and the stop-gradient operation. Although SimSiam is effective with small bath sizes, it is not well understood, and hard to interpret its architectural tricks [10]. In contrast, MVEB can work well without architectural tricks and the requirement for a large batch size.

7.2 Target Branch Type

The self-supervised learning methods with Siamese networks adopt different types of the target branch. We select two common types for it: weight-sharing and momentum-update. In SimCLR [2], two branches share the same weights and are updated simultaneously, which is referred to as a symmetric network. MoCo [3] uses the momentum encoder as the target branch, which performs momentum updates according to the other branch.

We use the same projector network for both weight-sharing and momentum-update branches. Specifically, the projector network consists of three linear layers, each with the output dimensionality set to 8194. We apply BN and ReLU after each of the first two layers. The predictor network is not used in Siamese networks.

Weight-Sharing Branch. The batch size is set to 1024. Other configurations are kept the same as the pre-training setting in Section 7.1.

Momentum-Update Branch. We train 100 epochs with the SGD optimizer. The weight decay and the momentum are set to $1e-4$ and 0.9, respectively. The basic learning rate is 0.1, scaled with the batch size and divided by 256. We decrease the learning rate to one-thousandth of it with the cosine decay scheduler after a warm-up period of 5 epochs. The loss function coefficient β is set to 0.01. The batch size is 1024. Following the setting of BYOL [6], we update the target branch with the exponential moving by increasing the average parameters from 0.996 to 1 with the cosine scheduler.

We empirically study the effect of these two types for MVEB. For the weight-sharing target branch, the linear evaluation on ImageNet is 68.9%. In contrast, the momentum encoder can achieve 71.2% with the linear evaluation on ImageNet. This shows that MVEB is more beneficial with the momentum encoder. Hence, we adopt it as the target branch in our experiments.

7.3 Loss Balance Coefficient

The objective function of MVEB in Eq. (11) consists of two terms, each having different roles. The first term $\mathbb{E}_{p(\mathbf{z}_1, \mathbf{z}_2)}[\mathbf{z}_2^T \mathbf{z}_1]$ learns invariant representation for different views of a sample. Maximizing the second term $H(\mathbf{z}_1) + H(\mathbf{z}_1)$ increases the uniformity of the embeddings.

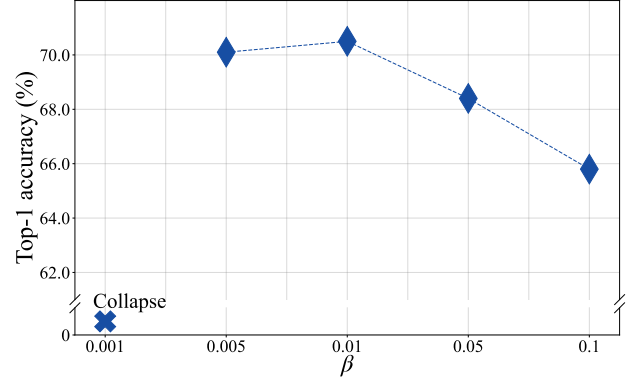


Fig. 4. Linear classification on ImageNet by MVEB pretrained with different coefficients β . Collapse means that the accuracy of the linear classification is 0.

TABLE 7
Top-1 accuracies (%) of linear classification on ImageNet [30] with ViT-s and ResNet-50. The bold entries denote the best.

Top-1 Method	Backbone	
	ResNet-50 (23M)	ViT-s (21M)
SimCLR [2]	69.3	69.0
SwAV [19]	71.8	67.1
BYOL [6]	74.3	71.0
MoCo V3 [58]	73.8	72.5
DINO [8]	-	70.9
MAE [36]	-	68.2
MVEB (ours)	74.6	73.4

We study its importance and report the performance in Fig. 4. We can observe that all the representations collapse to a constant vector when β is 0.001. Since β approaches zero, the Siamese network suffers from model collapse with the trivial constant representations without maximizing uniformity. In Eq. (10), $\beta = \frac{1}{(\lambda+1)\kappa}$, where κ is a constant. As shown in Eq. (1), λ is the coefficient to balance the optimization of $I_\phi(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ and $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$. When increasing the value of β , which decreases λ , the objective pays more attention to maximizing $I_\phi(\mathbf{z}_1; \mathbf{v}_2)$ to keep the information relevant to $\mathbf{v}_2$. However, the performance of the model degrades when β is greater than 0.01. This is because increasing β to make λ smaller than a threshold does not eliminate superfluous information effectively, which hurts the performance of downstream tasks.

7.4 Generalization Across Different Backbones

In this section, we evaluate the capability of our approach to generalize effectively across both ViTs and ConvNets. We follow the experiment setting on ImageNet in [58], and compare MVEB with previous popular SSL methods based on the Siamese network and MAE [36] across both ViTs and ConvNets in Table 7. The results indicate that MVEB demonstrates competitive performance with both ViTs and ConvNets.

7.5 Pretraining Efficiency

To evaluate the pretraining efficacy of the MVEB approach, we conduct a comparative analysis against two established methods: BYOL and Barlow Twins. To ensure an equitable

evaluation, we standardize the experimental setup by employing a Resnet50 architecture as the underlying backbone. We assess the pretraining duration required by each method to complete 1000 epochs with a batch size of 4096 on ImageNet. The experimental configurations for BYOL and Barlow Twins are aligned with the parameters detailed in their respective seminal papers [6] and [9].

MVEB takes approximately 81 hours to distribute on 32 NVIDIA V100 GPUs. The reimplementations of BYOL and Barlow Twins take approximately 89 and 80 hours on the same hardware with the same setting, respectively. Hence, the pretraining cost of MVEB is similar to those of BYOL and Barlow Twins.

7.6 Comparison of MVEB with MVIB

The work in [15] conducts analytical experiments on CIFAR10 to compare MVIB with SimCLR. The linear evaluation accuracy of SimCLR is 85.76%, while the linear evaluation accuracy of MVIB is 86.2%. This indicates that MVIB cannot improve its performance much compared with SimCLR. We follow the same experiment setting as [15] on CIFAR10 to compare MVEB with MVIB. The linear evaluation accuracy of MVEB is 90.42%. This accuracy surpasses that of MVIB. Besides, MVEB can be directly applied to the Siamese network, but MVIB requires an additional stochastic net to obtain the feature distribution as shown in Fig. 3.

8 DISCUSSION AND CONCLUSION

The core of self-supervised learning is that the learned representation can generalize well to downstream tasks. The minimal sufficient representation can improve the generalization. We propose the multi-view entropy bottleneck (MVEB), a novel pretext task to learn the minimal sufficient representation. It can be further simplified to maximizing both the agreement between the embeddings of the two views of a sample and the differential entropy of the embedding distribution. We present the score-based entropy estimator with the von Mises-Fisher kernel to approximate the gradient of the differential entropy. This estimator can be used to maximize the differential entropy to prevent collapse efficiently. Extensive experiments show that MVEB generalizes well across various downstream tasks and establishes new state-of-the-art results.

Limitation. Existing self-supervised methods with Siamese networks are based on the common assumption in multi-view learning: either view is (approximately) sufficient for the prediction of downstream tasks and contains the same task-relevant information. Hence, the non-shared task-relevant information between the views can be ignored. Our experiment results also verify that the minimal sufficient representation can improve the generalization for downstream tasks. If the discrepancy between the two views is too large, the non-shared task-relevant information cannot be ignored. In other words, either view is not sufficient for the prediction of downstream tasks. Thus either view cannot be regarded as a supervised signal to extract task-relevant information and eliminate the superfluous information in Siamese networks. How to overcome this problem is future work.

REFERENCES

- [1] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018.
- [2] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [3] K. He, H. Fan, Y. Wu, S. Xie, and R. B. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [4] X. Wang and G.-J. Qi, "Contrastive learning with stronger augmentations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 5549–5560, 2023.
- [5] Y. Wang, J. Lin, Q. Cai, Y. Pan, T. Yao, H. Chao, and T. Mei, "A low rank promoting prior for unsupervised contrastive learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 2667–2681, 2023.
- [6] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - A new approach to self-supervised learning," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020.
- [7] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [8] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [9] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction," in *Proc. Int. Conf. Mach. Learn.*, 2021.
- [10] A. Bardes, J. Ponce, and Y. LeCun, "Vicreg: Variance-invariance-covariance regularization for self-supervised learning," in *Proc. Int. Conf. Learn. Representations*, 2022.
- [11] K. Sridharan and S. M. Kakade, "An information theoretic framework for multi-view learning," in *Annual Conference on Learning Theory*, 2008.
- [12] M. Federici, A. Dutta, P. Forré, N. Kushman, and Z. Akata, "Learning robust representations via multi-view information bottleneck," in *Proc. Int. Conf. Learn. Representations*, 2020.
- [13] Y. H. Tsai, Y. Wu, R. Salakhutdinov, and L. Morency, "Self-supervised learning from a multi-view perspective," in *Proc. Int. Conf. Learn. Representations*, 2021.
- [14] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," *arXiv preprint physics/0004057*, 2000.
- [15] H. Wang, X. Guo, Z.-H. Deng, and Y. Lu, "Rethinking minimal sufficient representation in contrastive learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [16] G. L. Gaile and J. E. Burt, "Directional statistics (2nd edition)." 2020.
- [17] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. A. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1734–1747, 2016.
- [18] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. Eur. Conf. Comput. Vis.*, 2018.
- [19] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020.
- [20] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. Int. Conf. Mach. Learn.*, 2020.
- [21] S. Zhang, F. Zhu, J. Yan, R. Zhao, and X. Yang, "Zero-cl: Instance and feature decorrelation for negative-free symmetric contrastive learning," in *Proc. Int. Conf. Learn. Representations*, 2022.
- [22] A. Ermolov, A. Siarohin, E. Seginov, and N. Sebe, "Whitening for self-supervised representation learning," in *Proc. Int. Conf. Mach. Learn.*, 2021.
- [23] A. Achille and S. Soatto, "Emergence of invariance and disentanglement in deep representations," *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 1947–1980, 2018.
- [24] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *Adv. Neural Inform. Process. Syst.*, 2018.

- [25] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, "Mine: mutual information neural estimation," *Proc. Int. Conf. Mach. Learn.*, 2018.
- [26] L. Wen, Y. Zhou, L. He, M. Zhou, and Z. Xu, "Mutual information gradient estimation for representation learning," in *Proc. Int. Conf. Learn. Representations*, 2020.
- [27] G. Roeder, Y. Wu, and D. K. Duvenaud, "Sticking the landing: Simple, lower-variance gradient estimators for variational inference," in *Proc. Adv. Neural Inform. Process. Syst.*, 2017.
- [28] Y. Li and R. E. Turner, "Gradient estimators for implicit models," in *Proc. Int. Conf. Learn. Representations*, 2018.
- [29] C. Tao, H. Wang, X. Zhu, J. Dong, S. Song, G. Huang, and J. Dai, "Exploring the equivalence of siamese self-supervised learning via a unified gradient framework," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022.
- [30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009.
- [31] Y. You, I. Gitman, and B. Ginsburg, "Scaling sgd batch size to 32k for imagenet training," *arXiv preprint arXiv:1708.03888*, 2017.
- [32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016.
- [33] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.
- [34] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola, "What makes for good views for contrastive learning?" in *Proc. Adv. Neural Inform. Process. Syst.*, 2020.
- [35] E. Amrani and A. Bronstein, "Self-supervised classification network," in *Proc. Eur. Conf. Comput. Vis.*, 2022.
- [36] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick, "Masked autoencoders are scalable vision learners," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 15979–15988.
- [37] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu, "Simim: a simple framework for masked image modeling," in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022, pp. 9643–9653.
- [38] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis.*, 2016.
- [39] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4l: Self-supervised semi-supervised learning," in *Proc. Int. Conf. Comput. Vis.*, 2019.
- [40] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.
- [41] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Worksh.*, 2004.
- [42] J. Krause, J. Deng, M. Stark, and L. Fei-Fei, "Collecting a large-scale dataset of fine-grained cars," in *Proc. Second Workshop on Fine-Grained Visual Categorization*, 2013.
- [43] A. Krizhevsky, G. Hinton et al., "Learning multiple layers of features from tiny images," 2009. [Online]. Available: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>
- [44] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014.
- [45] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proc. Indian Conference on Computer Vision, Graphics & Image Processing*, 2008.
- [46] L. Bossard, M. Guillaumin, and L. V. Gool, "Food-101-mining discriminative components with random forests," in *Proc. Eur. Conf. Comput. Vis.*, 2014.
- [47] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "Cats and dogs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012.
- [48] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010.
- [49] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *Int. J. Comput. Vis.*, vol. 88, pp. 303–338, 2010.
- [50] Y. Xiao, Z. Wei, and Z. Wang, "A limited memory bfgs-type method for large-scale unconstrained optimization," *Computers & Mathematics with Applications*, vol. 56, no. 4, pp. 1001–1009, 2008.
- [51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inform. Process. Syst.*, 2019.
- [52] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," in *Proc. Adv. Neural Inform. Process. Syst.*, 2020.
- [53] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense contrastive learning for self-supervised visual pre-training," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021.
- [54] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014.
- [55] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017.
- [56] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," <https://github.com/facebookresearch/detectron2>, 2019.
- [57] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [58] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," in *Int. Conf. Comput. Vis.*, 2021.



Liangjian Wen is currently an assistant professor with the School of Computing and Artificial Intelligence, Southwestern University of Finance and Economics, Chengdu, China. I obtained my Ph.D. degree from School of Computer Science and Engineering, University of Electronic Science and Technology of China (UESTC) in July 2021. His research interests include machine learning, deep learning, representation learning, and self-supervised learning.



Xiasi Wang is a Ph.D. student majoring in IIP (Data Science and Analytics) at Academy of Interdisciplinary Studies, the Hong Kong University of Science and Technology, and he is supervised by Prof. Yuan Yao and Prof. Nevin L. Zhang. He received the Bachelor degree of Statistics from the School of Management at University of Science and Technology of China in 2020. His research interest lies in representation learning.



Jianzhuang Liu (Senior Member, IEEE) received the PhD degree in computer vision from The Chinese University of Hong Kong, in 1997. From 1998 to 2000, he was a research fellow with Nanyang Technological University, Singapore. From 2000 to 2012, he was a post-doctoral fellow, an assistant professor, and an adjunct associate professor with The Chinese University of Hong Kong, Hong Kong. In 2011, he joined the Shenzhen Institute of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen, China, as a professor. He was a principal researcher in Huawei Company from 2012 to 2023. He has authored more than 200 papers in the areas of computer vision, image processing, deep learning, and AIGC.



Zenglin Xu received a Ph.D. degree in computer science and engineering from the Chinese University of Hong Kong. He is currently a full professor at the Harbin Institute of Technology Shenzhen, and also affiliated with Peng Cheng Lab. He has been with Michigan State University, the Cluster of Excellence at Saarland University and the Max Planck Institute for Informatics, Purdue University, and the University of Electronic Science and Technology of China. Dr. Xu's research interests include machine learning and its applications in computer vision, health informatics, and natural language processing. He is the recipient of the outstanding student paper honorable mention at AAAI 2015, the best student paper runner up at ACM 2016, and the 2016 young researcher award from APNNS. He serves as an associate editor of Neural Networks. He is currently the vice president for education in the International Neural Network Society.

APPENDIX

A Stein Gradient Estimator

A promising method for estimating the score of an implicit distribution is the Stein gradient estimator proposed in [28]. Here we briefly describe it.

Assume $\mathbf{x}$ is supported on $\mathcal{X} \subseteq \mathbb{R}^d$. Let $q(\mathbf{x})$ be a continuously differentiable probability density function, and $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_{d'}(\mathbf{x})]^T$ be a d' -dimensional differentiable vector function, where the following boundary condition is satisfied:

$$q(\mathbf{x})\mathbf{h}(\mathbf{x}) = \mathbf{0}, \forall \mathbf{x} \in \partial\mathcal{X} \text{ if } \mathcal{X} \text{ is compact, or} \\ \lim_{\mathbf{x} \rightarrow \infty} q(\mathbf{x})\mathbf{h}(\mathbf{x}) = \mathbf{0} \text{ if } \mathcal{X} = \mathbb{R}^d. \quad (22)$$

We call $\mathbf{h}(\mathbf{x})$ the Stein class of $q(\mathbf{x})$ if the above condition holds. Then the following Stein's identity can be derived by using integration by parts:

$$\mathbb{E}_q [\mathbf{h}(\mathbf{x})[\nabla_{\mathbf{x}} \log q(\mathbf{x})]^T + \nabla_{\mathbf{x}} \mathbf{h}(\mathbf{x})] = \mathbf{0}, \quad (23)$$

where $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ is the score of $q(\mathbf{x})$.

The Monte Carlo method can be adopted to estimate the expectation in Eq. (23), which establishes a connection between samples from $q(\mathbf{x})$ and the score $\nabla_{\mathbf{x}} \log q(\mathbf{x})$. Let $\mathbf{x}^{1:M}$ be M i.i.d. samples drawn from $q(\mathbf{x})$. Monte Carlo sampling shows:

$$-\frac{1}{M} \mathbf{H} \mathbf{G} \approx \overline{\nabla_{\mathbf{x}} \mathbf{h}}, \quad (24)$$

where $\mathbf{H} = [\mathbf{h}(\mathbf{x}^1), \dots, \mathbf{h}(\mathbf{x}^M)] \in \mathbb{R}^{d' \times M}$, $\mathbf{G} = [\nabla_{\mathbf{x}^1} \log q(\mathbf{x}^1), \dots, \nabla_{\mathbf{x}^M} \log q(\mathbf{x}^M)]^T \in \mathbb{R}^{M \times d}$, $\overline{\nabla_{\mathbf{x}} \mathbf{h}} = \frac{1}{M} \sum_{m=1}^M \nabla_{\mathbf{x}^m} \mathbf{h}(\mathbf{x}^m) \in \mathbb{R}^{d' \times d}$ and $\nabla_{\mathbf{x}^m} \mathbf{h}(\mathbf{x}^m) = [\nabla_{\mathbf{x}^m} h_1(\mathbf{x}^m), \dots, \nabla_{\mathbf{x}^m} h_{d'}(\mathbf{x}^m)]^T \in \mathbb{R}^{d' \times d}$. This motivates the ridge regression problem:

$$\underset{\hat{\mathbf{G}} \in \mathbb{R}^{M \times d}}{\text{argmin}} \left\| \overline{\nabla_{\mathbf{x}} \mathbf{h}} + \frac{1}{M} \mathbf{H} \hat{\mathbf{G}} \right\|_F^2 + \frac{\eta}{M^2} \|\hat{\mathbf{G}}\|_F^2, \quad (25)$$

where $\|\cdot\|_F^2$ denotes the Frobenius norm of a matrix and $\eta > 0$ is the regularization coefficient. An analytic solution of Eq.(25) is:

$$\hat{\mathbf{G}}^{\text{Stein}} = -M(\mathbf{K} + \eta \mathbf{I})^{-1} \mathbf{H}^T \overline{\nabla_{\mathbf{x}} \mathbf{h}}, \quad (26)$$

where $\mathbf{K} = \mathbf{H}^T \mathbf{H}$. We rewrite $\mathbf{K}_{ij} = \mathbf{h}(\mathbf{x}^i)^T \mathbf{h}(\mathbf{x}^j)$ by defining a positive definite kernel $k : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ so that $\mathbf{K}_{ij} = k(\mathbf{x}^i, \mathbf{x}^j)$. Similarly, we have $(\mathbf{H}^T \overline{\nabla_{\mathbf{x}} \mathbf{h}})_{ij} = \frac{1}{M} \sum_{m=1}^M \nabla_{\mathbf{x}_j^m} k(\mathbf{x}^i, \mathbf{x}^m)$. In this way, $\hat{\mathbf{G}}^{\text{Stein}}$ and the estimation of the score $\nabla_{\mathbf{x}} \log q(\mathbf{x})$ can be obtained.

B Proof of Eq. (3): $I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) = H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2)$

$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ is defined as follows:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) = \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \\ - \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) p_{\phi}(\mathbf{v}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \quad (27)$$

The first term on the right-hand side of Eq. 27 can be decomposed as follows:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \\ + \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{v}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1, \quad (28)$$

and the second term can be decomposed as:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \\ + \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{v}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1. \quad (29)$$

Hence, $I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2)$ is represented as:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) = \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \\ - \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 \quad (30)$$

The first term on the right-hand side of Eq. 30 can be rewritten as:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 = -H(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2), \quad (31)$$

and the second term can be derived as :

$$- \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) d\mathbf{z}_1 d\mathbf{v}_1 = \\ - \int p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_2) d\mathbf{z}_1 = H(\mathbf{z}_1 | \mathbf{v}_2).$$

Finally, we obtain:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1 | \mathbf{v}_2) = H(\mathbf{z}_1 | \mathbf{v}_2) - H(\mathbf{z}_1 | \mathbf{v}_1, \mathbf{v}_2). \quad (32)$$

C Proof of Eq. (4): $I_{\phi}(\mathbf{z}_1; \mathbf{v}_1) = H(\mathbf{z}_1) - H(\mathbf{z}_1 | \mathbf{v}_1)$

$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1)$ is defined as follows:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1) = \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1 \\ - \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1) p_{\phi}(\mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1. \quad (33)$$

The first term on the right-hand side of Eq. 33 can be decomposed as follows:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1 \\ + \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1, \quad (34)$$

and the second term can be decomposed as:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1) d\mathbf{z}_1 d\mathbf{v}_1 \\ + \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1. \quad (35)$$

Hence, $I_{\phi}(\mathbf{z}_1; \mathbf{v}_1)$ is represented as:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1) = \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1 \\ - \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1) d\mathbf{z}_1 d\mathbf{v}_1. \quad (36)$$

The first term on the right-hand side of Eq. 36 can be rewritten as:

$$\int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1 | \mathbf{v}_1) d\mathbf{z}_1 d\mathbf{v}_1 = -H(\mathbf{z}_1 | \mathbf{v}_1), \quad (37)$$

and the second term can be derived as :

$$\begin{aligned} & - \int p_{\phi}(\mathbf{z}_1, \mathbf{v}_1) \log p_{\phi}(\mathbf{z}_1) d\mathbf{z}_1 d\mathbf{v}_1 = \\ & - \int p_{\phi}(\mathbf{z}_1) \log p_{\phi}(\mathbf{z}_1) d\mathbf{z}_1 = H(\mathbf{z}_1). \end{aligned}$$

Finally, we obtain:

$$I_{\phi}(\mathbf{z}_1; \mathbf{v}_1) = H(\mathbf{z}_1) - H(\mathbf{z}_1 | \mathbf{v}_1). \quad (38)$$