ESCo: Towards Provably Effective and Scalable Contrastive Representation Learning

Anonymous authors
Paper under double-blind review

Abstract

InfoNCE-based contrastive learning models (e.g., MoCo, SimCLR, etc.) have shown inspiring power in unsupervised representation learning by maximizing a tight lower bound of the mutual information of two views' representations. However, its quadratic complexity makes it hard for scaling to larger batch sizes, and some recent research suggests that it may exploit superfluous information that is useless for downstream prediction tasks. In this paper, we propose ESCo (**E**ffective and **S**calable **Co**ntrastive), a new contrastive framework which is essentially an instantiation of the Information Bottleneck principle under self-supervised learning settings. Specifically, ESCo targets a new objective that seeks to maximize the similarity between the representations of positive pairs and minimize the pair-wise kernel potential of negative pairs, with a provable guarantee of *effective* representations that preserve task-relevant information and discard the irrelevant one. Furthermore, to escape from the quadratic time complexity and memory cost, we propose to leverage the Random Features to achieve accurate approximation with linear *scalability*. We show that the vanilla InfoNCE objective is a degenerated case of ESCo, which implies that ESCo can potentially boost existing InfoNCE-based models. To verify our method, we conduct extensive experiments on both synthetic and real-world datasets, showing its superior performance over the InfoNCE-based baselines in (unsupervised) representation learning tasks for images and graphs.

1 Introduction

Contrastive learning (Hjelm et al., 2019; van den Oord et al., 2018; Belghazi et al., 2018) has achieved remarkable success in unsupervised representation learning without using costly handcrafted labels. Currently, the state-of-the-art contrastive models follow a multi-view perspective: two views of the input data are first generated using random augmentations, and then their representations are trained using contrastive loss. The key insight is to discriminate positive pairs (the views generated from the same input) from negative ones (the views of distinct data points) in order construct self-supervised signals. Among the contrasting methods, InfoNCE objective (van den Oord et al., 2018) based models (e.g. CMC (Tian et al., 2020a), MoCo (He et al., 2020), SimCLR (Chen et al., 2020), etc.) have become the most popular ones. Formally, under multi-view settings, one has two sets of data points, $\{\boldsymbol{x}_1^A, \cdots, \boldsymbol{x}_N^A\}$ from view A and $\{\boldsymbol{x}_1^B, \cdots, \boldsymbol{x}_N^B\}$ from view B, and the InfoNCE approach learns an encoder $f(\cdot)$ by optimizing the objective:

$$\mathcal{L}_A = \sum_{i=1}^{N} -\log \frac{e^{f(\boldsymbol{x}_i^A)^\top f(\boldsymbol{x}_j^B)/\tau}}{\sum_{j=1}^{N} e^{f(\boldsymbol{x}_i^A)^\top f(\boldsymbol{x}_j^B)/\tau}} \quad (\mathcal{L}_B \text{ symmetrically}) \tag{1}$$

The foundation of InfoNCE lies in its connection with the Information Theory. Denote the random variables of the two views by X_A and X_B respectively, and then minimizing Eq. 1 is to maximize a tight variational lower bound of $I(f(X_A), f(X_B))$. This makes the learned representations keep the shared information as much as possible, which presumably should be informative for downstream tasks (Tsai et al., 2021).

Despite its theoretical grounds and promising results, InfoNCE-based models suffer from its quadratic complexity with respect to the number of negative samples ($\mathcal{O}(N^2)$), since a large quantity of negative samples are required to ensure precise approximation of the denominator in Eq. 1. While some up-to-date works (Grill et al., 2020; Chen & He, 2020) propose effective remedy with linear complexity,

the key rationales behind their success still remain unclear (Tian et al., 2021; Richemond et al., 2020). Moreover, recent studies (Federici et al., 2020; Tsai et al., 2021; Zhang et al., 2021) suggest that the encoder's representations may contain superfluous information that is useless for downstream predictions, which compromises the quality of the learned representations.

In this paper, we propose Effective and Scalable Contrastive Learning (ESCo), a novel framework for multi-view contrastive learning. Inspired by the recent progress in Multi-view Information Bottleneck (MIB) (Federici et al., 2020; Tsai et al., 2021), ESCo is derived from an instantiation of the IB principle under multi-view self-supervised learning setting. More specifically, the objective of MIB can be converted into a weighted combination of 1) a conditional entropy minimization term and 2) an entropy maximization term. The first one can be achieved by minimizing a simple MSE loss between the representations of two views, while the second one could be realized by minimizing the total pairwise potential w.r.t. a certain kernel function. Such a formulation guarantees that the ideal solution induces *effective* learned representations that preserve all the task-relevant information and discard as much the irrelevant one as possible. On top of this, to reduce the quadratic complexity for kernel computation, we leverage Random Fourier Features (RFF) (Rahimi et al., 2007) and its improved variant Structured Orthogonal Random Features (SORF) (Yu et al., 2016) to accurately approximate the kernel function with linear complexity. Such a design endows the approach with desirable *scalability* to large-scale training samples.

To shed more insights on the rationale of the proposed approach, we show that the vanilla InfoNCE objective is essentially a degenerated case of ESCo with Gaussian kernels, which indicates that ESCo can be potentially applied to enhance off-the-shelf InfoNCE-based models. The linear complexity of ESCo also paves the way for harnessing a much larger quantity of negative samples with computation budget controlled within the same level, which can indeed further strengthen the *effectiveness* of InfoNCE as a lower bound surrogate of the mutual information objective. We apply our approach to a variety of practical tasks, including synthetic data and real-world data spanning from images to graphs. The results demonstrate that compared with InfoNCE-based counterparts, ESCo yields superior performance with much less memory and time cost. Furthermore, we evaluate the model with various dataset sizes, and demonstrate its applicability to large-scale datasets (batch sizes) without large memory/time costs. **The highlights of this paper can be summarized as follows**:

1) We propose ESCo, a novel contrastive framework for unsupervised representation learning. For its effectiveness, ESCo is an effective embodiment of Multi-view Information Bottleneck. Compared with recent emerging methods that target MIB, ESCo does not require additional self-supervised learning tasks (such as inverse predictive learning), and is versatile for a rich family of kernel functions. We also show that ESCo possesses potential for boosting existing contrastive models.

2) For scalability, we take advantage of Random Fourier Features (RFF) and Structured Orthogonal Random Features (SORF) to reduce the space-time complexity of ESCo from quadratic to linear, enabling it to harness more negative samples without compromising efficiency and thus can be trained with much larger batch sizes.

3) We conduct thorough experiments on synthetic datasets and also demonstrate the power of ESCo in real-world tasks that require representation learning for images and graphs. The results show that ESCo outperforms its baseline counterparts, with much less GPU memory cost and training time.

2 Related Works and Preliminaries

2.1 Contrastive Self-supervised Learning

Self-Supervised Learning (SSL) aims at learning informative representations without the availability of labels through well-defined auxiliary tasks. Among current SSL models, multi-view contrastive methods have achieved state-of-the-art performance in self-supervised representation learning (van den Oord et al., 2018; Tian et al., 2020a; He et al., 2020; Chen et al., 2020). These approaches first generate two views of the same input data through random augmentations, and then use contrastive loss (typically the InfoNCE loss (van den Oord et al., 2018)) to maximize a tight lower bound of the mutual information between the two views (Tschannen et al., 2019; Poole et al., 2019).

Despite its theoretical soundness and promising empirical performance on various tasks, InfoNCE-based contrastive learning suffers from its quadratic complexity w.r.t. the number of data samples, given the fact that it requires a large number of negative samples to ensure that the mutual infor-

mation lower bound is tight enough (Poole et al., 2019). To explore negative-sample-free methods, BYOL (Grill et al., 2020) and Simsiam (Chen & He, 2020) adopt asymmetric structures to avoid trivial degenerated solutions without using negative samples. Although a recent study (Tian et al., 2021) sheds some lights on understanding of their success, the rationales still remain uncleared especially for why such methods could learn informative representations. Other works seek to detour negative sampling through feature-level decorrelation (Zbontar et al., 2021; Bardes et al., 2021), yet these models resort to assumptions for data distributions (e.g., Gaussian forms), which could be violated in real-world scenarios.

2.2 Interpreting Self-supervised Learning with Information Theory

As mentioned above, the success of contrastive learning is often attributed to maximizing the mutual information between the representations of two views. Tsai et al. (2021) points out that most self-supervised frameworks, either predictive (Devlin et al., 2019; Noroozi & Favaro, 2016) or contrastive, can be viewed as maximizing the mutual information between representations of input data and pre-defined self-supervised signals. Inspired by the Information Bottleneck (IB) principle in supervised representation learning (Tishby et al., 2000; Tishby & Zaslavsky, 2015; Strouse & Schwab, 2016), some recent studies generalize the spirits to multi-view self-supervised learning and proposes Multi-view Information Bottleneck (MIB) (Tsai et al., 2021; Federici et al., 2020; Zhang et al., 2021; Zbontar et al., 2021). MIB suggests that purely maximizing the mutual information between the representations (as is done by InfoNCE loss) may lead the learned representations to exploit redundant information that is useless for downstream predictions, so that an additional conditional entropy minimization term is required to fulfill the MIB principle (Tsai et al., 2021).

2.3 Kernel Approximation with Random Fourier Features

Random Fourier Features (RFF or Random Features in short) (Rahimi et al., 2007; Liu et al., 2020) is an effective technique for overcoming the poor scalability of kernel methods such as kernel SVM and kernel ridge regression (Avron et al., 2017) and kernel Independence test (Zhang et al., 2018; Li et al., 2021). Also, recently, RFF has been adopted to develop linear Transformers by approximating the softmax attention (Choromanski et al., 2020; Peng et al., 2021). Given d-dimensional vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ and a shift-invariant kernel $\kappa(\cdot)$, RFF constructs an explicit mapping Ψ: $\mathbb{R}^d \to \mathbb{R}^D$, such that $\kappa(\boldsymbol{x}, \boldsymbol{y}) \approx \Psi(\boldsymbol{x})^\top \Psi(\boldsymbol{y})$, which reduces the quadratic computation cost of the kernel matrix to a linear one w.r.t data size. Generally, given a positive definite shift-invariant kernel $\kappa(\boldsymbol{x}, \boldsymbol{y}) = \kappa(\boldsymbol{x} - \boldsymbol{y})$, the Fourier transform p of kernel κ is $p(\boldsymbol{\omega}) = \frac{1}{2\pi} \int e^{-j\boldsymbol{\omega}'\Delta} k(\Delta) \mathrm{d}\Delta$, where $\Delta = \boldsymbol{x} - \boldsymbol{y}$. Then we could draw D i.i.d. samples $\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_D \in \mathbb{R}^d$ from p, and $\Psi(\boldsymbol{x})$ is represented as:

$$\Psi(\boldsymbol{x}) = \sqrt{\frac{1}{D}} \left[\cos(\boldsymbol{\omega}_1^\top \boldsymbol{x}), \cdots, \cos(\boldsymbol{\omega}_D^\top \boldsymbol{x}), \sin(\boldsymbol{\omega}_1^\top \boldsymbol{x}), \cdots, \sin(\boldsymbol{\omega}_D^\top \boldsymbol{x})\right]^\top. \tag{2}$$

Let $\mathbf{W} = [\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_D] \in \mathbb{R}^{d \times D}$ be a linear transformation matrix, one may realize that $\mathbf{W}^\top \boldsymbol{x}$ plays a central role in the above computation. Specifically, when $\kappa(\cdot)$ is a standard Gaussian kernel (i.e., RBF kernel), each entry of $\mathbf{W}$ can be directly sampled from a standard Gaussian distribution. The improved variants of RFF mainly concentrate on different ways to build the transformation matrix $\mathbf{W}$, so as to further reduce the computational complexity (Le et al., 2013) or lower the approximation variance (Yu et al., 2016).

3 Understanding Self-supervised Learning with Multi-View Information Bottleneck

In this section we first introduce Multi-view Information Bottleneck (MIB), which is the theoretical foundation of our proposed method. Then we provide an example to illustrate that vanilla contrastive learning objectives (i.e. InfoNCE) would fail to meet the MIB principle, which helps to convey the motivation of our method.

3.1 Multi-View Information Bottleneck

In supervised representation learning, the Information Bottleneck (IB) principle provides a principled interpretation for what kind of representations are optimal for a specific task. Denote the random

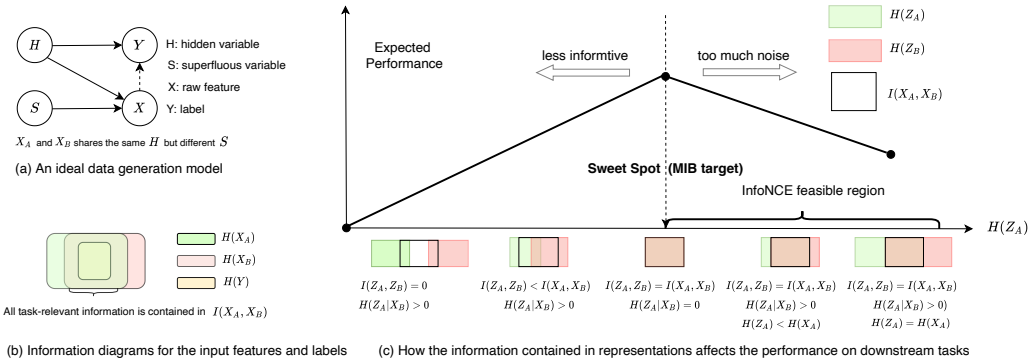

Figure 1: (a) An ideal data generation process that follows the multi-view redundancy assumption. (b) When the data generation process in (a) holds, all the task-relevant information $H(Y)$ is contained in the shared part between two views – $I(X_A, X_B)$. (c) An illustration for the impact of the information contained in the embeddings on the performance of downstream tasks. Zoom in for better view.

variables of input data, low-dimensional embeddings, and labels as X, Z and Y, respectively. IB formulates representation learning task as a constrained optimization problem:

$$\min I(Z, X) \text{ s.t. } I(Z, Y) \text{ is maximized}, \tag{3}$$

where $I(\cdot, \cdot)$ denotes the mutual information between two random variables, and β is a Lagrange multiplier. The implication of IB is that the optimal representations should contain maximal information that is useful for prediction, and meanwhile compress the input data as much as possible. The Multi-view Information Bottleneck is defined in a similar way.

Definition 1. *(Multi-View Information Bottleneck (Federici et al., 2020; Tsai et al., 2021) aims at optimizing the following objective:*

$$\min I(Z_A, X_A) \textit{ s.t. } I(Z_A, X_B) \textit{ is maximized}, \tag{4}$$

where X_A, X_B denote the random variables of two views of input data, and Z_A, Z_B denote the random variables of their representations. The effectiveness of MIB for downstream tasks is guaranteed by the Multi-view Redundancy Assumption (Sridharan & Kakade, 2008), which assumes that almost all the task-relevant information is contained in the shared part between two views. Tsai et al. (2021) further prove that when this assumption holds, the learned representations through MIB keeps maximal task-relevant information and discard as much task-irrelevant information as possible, which provides an understanding for the rationale behind multi-view self-supervised learning.

As the constrained optimization problem in Eq. 1 is hard to solve with gradient descent, its relaxed form using Lagrangian relaxation is usually considered (Zbontar et al., 2021; Zhang et al., 2021):

$$\begin{aligned} &\min \quad I(Z_A, X_A) + \beta I(Z_A, X_B) \\ \text{or } &\min \quad (\beta/\beta - 1) H(Z_A|X_B) - H(Z_A), \text{ where } \beta/(\beta - 1) > 1. \end{aligned} \tag{5}$$

Here $H(\cdot)$ and $H(\cdot|\cdot)$ denote entropy and conditional entropy, respectively. Note that in Eq. 4 and Eq. 5 we treat view A as the primal view, and we have a symmetric objective for view B. For convenience, in the following presentation, we only consider view A as the primal view, and the final loss function is the combination of both views.

3.2 The Redundancy Issue of Contrastive Learning

Recent studies (Tian et al., 2020b; Tsai et al., 2021) point out that the ideal representations through multi-view learning are expected to preserve the shared information between (two) views and in the meanwhile discard as much redundant information (not shared across views) as possible. Yet, current contrastive learning methods (e.g. InfoNCE) would fail to achieve the second target. We provide an illustration in Fig. 1 through a toy example.

We consider an ideal data generation process of multi-view data, as shown in Fig. 1(a). We use H to denote a hidden variable, which decides the label of each data point, and S to denote a superfluous

variable, which is irrelevant to the label but decides the input data X together with H. In this example, all the task-relevant information is contained in the shared part between two views $I(X_A, X_B)$ (i.e., Fig. 1(b)). In Fig. 1(c), we show that the optimal representation should be informative enough ($I(Z_A, Z_B) = I(X_A, X_B)$), and contains no noise ($H(Z_A|X_B) = 0$). Unfortunately, contrastive learning approaches (e.g. InfoNCE loss) learn informative representations by purely maximizing $I(Z_A, Z_B)$, while the redundant information $H(Z_A|X_B)$ is out of control, which may compromise the quality of learned representations.

To refine contrastive learning with the ability of reducing superfluous information, Tsai et al. (2021) combine InfoNCE objective with other self-supervised learning tasks, while Tian et al. (2020b) resort to learnable data augmentations that aim to reduce the redundant information across views. This paper inherits the spirit in (Tsai et al., 2021) that reduces redundant information by minimizing the conditional entropy between two views, which will be detailedly presented in Sec. 4.

4 METHODOLOGY

Recall that the MIB objective in Eq. 5 consists of two terms: the conditional entropy term $H(Z_A|X_B)$ which should be minimized, and the entropy term $H(Z_A)$ which should be maximized. We next introduce how to reach the two targets respectively with empirical estimation.

Minimization of $H(Z_A|X_B)$. Given that $H(Z_A|X_B) = H(Z_A|Z_B) - I(Z_A, X_B|Z_B) \leq H(Z_A|Z_B)$, we can minimize $H(Z_A|Z_B)$ instead. We further adopt an assumption that the conditional distribution $p(Z_A|Z_B)$ is a Gaussian distribution with diagonal covariance $\mathcal{N}(Z_B, \sigma^2 \boldsymbol{I})$. Then the minimization of $H(Z_A|Z_B)$ could be achieved by simply minimizing the MSE loss between the representations of positive pairs (i.e., reconstruction loss) (Tsai et al., 2021):

$$\min \mathop{\mathbb{E}}_{(\boldsymbol{z}^A, \boldsymbol{z}^B)} \|\boldsymbol{z}^A - \boldsymbol{z}^B\|_2^2, \quad (\boldsymbol{z}^A, \boldsymbol{z}^B) \sim P_{Z_A, Z_B}. \tag{6}$$

Maximization of $H(Z_A)$. Note that in most contrastive learning models (He et al., 2020; Chen et al., 2020), the final outputs are l_2-normalized, so that both Z_A and Z_B are distributed on a unit hypersphere. Obviously we know that if Z_A and Z_B are distributed uniformly on the hypersphere, $H(Z_A)$ and $H(Z_B)$ would be maximized. The problem of enforcing points to be uniformly distributed on the unit hypersphere is a well-studied one in the literature (Wang & Isola, 2020), and can be realized by minimizing the total pairwise potential w.r.t. a certain kernel function (e.g. Gaussian kernel):

$$\min \log \mathop{\mathbb{E}}_{\boldsymbol{z}_1, \boldsymbol{z}_2} [\kappa(\boldsymbol{z}_1, \boldsymbol{z}_2)], \quad \boldsymbol{z}_1, \boldsymbol{z}_2 \overset{\text{i.i.d.}}{\sim} P_{Z_A}, \tag{7}$$

where $\kappa(\cdot, \cdot)$ is a kernel function (usuall should be shift-invariant, e.g. Gaussian kernel); $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are i.i.d. samples from P_{Z_A}. Combining Eq. 6 and 7, the MIB objective in Eq. 5 can be converted into the following one:

$$\min \lambda \mathbb{E}_{(\boldsymbol{z}^A, \boldsymbol{z}^B) \sim P_{Z_A, Z_B}} \|\boldsymbol{z}^A - \boldsymbol{z}^B\|_2^2 + \log \mathbb{E}_{\boldsymbol{z}_1, \boldsymbol{z}_2 \sim P_{Z_A}} [\kappa(\boldsymbol{z}_1, \boldsymbol{z}_2)], \tag{8}$$

where λ is a trade-off hyper-parameter. We remark that λ in Eq. 8 cannot be interpreted as $\beta/(\beta - 1)$ in Eq. 5, since Eq. 6 and Eq. 7 are not for estimation of mutual information. A follow-up question is how to instantiate Eq. 8 with empirical estimation, or in other words: 1) how to select the kernel function κ; 2) how to choose a proper λ so that MIB is achieved; 3) how to sample positive pairs and negative pairs. We next present a concrete implementation in Sec. 4.1 with Gaussian kernel, and then we will show its connection to the vanilla InfoNCE loss.

4.1 INSTANTIATION WITH GAUSSIAN KERNEL

Given a batch of data with N data points: $\{(\boldsymbol{x}_1^A, \boldsymbol{x}_1^B), \cdots, (\boldsymbol{x}_N^A, \boldsymbol{x}_N^B)\}$ and the corresponding l_2-normalized representations $\{(\boldsymbol{z}_1^A, \boldsymbol{z}_1^B), \cdots, (\boldsymbol{z}_N^A, \boldsymbol{z}_N^B)\}$, using Gaussian kernel $\kappa(\boldsymbol{x}, \boldsymbol{y}) = e^{-\|\boldsymbol{x}-\boldsymbol{y}\|_2^2/2\tau}$ and only using in-batch data points as negative samples, we can specify Eq. 8 as minimizing the loss function

$$\mathcal{L} = \sum_{i=1}^{N} \left(\lambda \|\boldsymbol{z}_i^A - \boldsymbol{z}_i^B\|_2^2 + \log \sum_{j=1}^{N} e^{-\|\boldsymbol{z}_i^A - \boldsymbol{z}_j^A\|_2^2/2\tau} \right). \tag{9}$$

When $\boldsymbol{x}$ and $\boldsymbol{y}$ are both l_2-normalized, we have $\|\boldsymbol{x}-\boldsymbol{y}\|_2^2 = 2 - 2\cdot\boldsymbol{x}^\top\boldsymbol{y}$ and we can further decompose the loss function of Eq. 9 into:

$$\mathcal{L} = \sum_{i=1}^{N}(-\log\frac{e^{{z_i^A}^\top z_i^B/\tau}}{\sum_{j=1}^{N} e^{{z_i^A}^\top z_j^A/\tau}} + (\lambda - \frac{1}{2\tau})\|z_i^A - z_i^B\|_2^2 + \text{const}). \tag{10}$$

Neglecting the constant term, the decomposed loss function in Eq. 10 is composed of two terms: 1) an InfoNCE-like term (by replacing z_j^B in the denominator with z_j^A), which plays the role of maximizing $I(Z_A, Z_B)$; 2) when $\lambda > 1/2\tau$, the MSE term aims at minimizing $H(Z_A|Z_B)$. Therefore, the combination of the two terms can achieve the goal of MIB on condition that $\lambda > 1/2\tau$.

Remark. Eq. 10 also reveals the connection between ESCo and InfoNCE objective: when $\kappa(\cdot,\cdot)$ is a Gaussian kernel $\kappa(\boldsymbol{x},\boldsymbol{y}) = e^{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2/2\tau}$ and $\lambda = \frac{1}{2\tau}$, our loss function in Eq. 9 would degenerate into InfoNCE in Eq. 1. This essential relationship suggests that our method can be naturally applied to combine with arbitrary InfoNCE-based models, which will be further discussed in Sec. 4.3.

Also, our approach is flexible for using other negative sample mining techniques such as memory bank (He et al., 2020) or use both intra-view and inter-view negative samples (Chen et al., 2020).

4.2 Fast and Accurate Kernel Approximation with Random Features

While we have shown that through selecting proper kernel function $\kappa(\cdot,\cdot)$ and trade-off hyper-parameter λ, the proposed objective function is capable for learning representations that meet the MIB principle, it still requires quadratic complexity to compute the pair-wise kernel functions. Observing that the pairwise kernel functions are summed up in Eq. 9, we can adopt the Random Feature technique as introduced in Sec. 2.3 to reduce the complexity of kernel computation:

$$\sum_{j=1}^{N}\kappa(z_i^A, z_j^A) \approx \sum_{j=1}^{N}\Phi(z_i^A)^\top\Phi(z_j^A) = \Phi(z_i^A)\sum_{j=1}^{N}\Phi(z_j^A), \tag{11}$$

where z_i^A and $z_j^A \in \mathbb{R}^d$ are input vectors, $\Phi(\boldsymbol{z}) = [\cos(\mathbf{W}^\top\boldsymbol{z}), \sin(\mathbf{W}^\top\boldsymbol{z})]/\sqrt{D}$ is the projection function, and $\mathbf{W}$ is a randomly generated transformation matrix. In particular, when using Gaussian kernel $\kappa(\boldsymbol{x},\boldsymbol{y}) = e^{-\|\boldsymbol{x}-\boldsymbol{y}\|_2^2/2\tau}$, the linear projection matrix can be obtained via $\mathbf{W} = \mathbf{G}/\sqrt{\tau} \in \mathbb{R}^{d\times D}$ where $G \in \mathbb{R}^{d\times D}$ is a random matrix with each entry sampled from standard Gaussian distribution $\mathcal{N}(0,1)$.

In fact, Random Fourier Feature (RFF) is an unbiased kernel estimator with space and time complexity of $\mathcal{O}(NDd)$ (N is the number of data points), which is linear w.r.t. the dataset size. However, the approximation variance is in inverse proportion to the projection dimension D, which means that it requires a large D to ensure the accuracy of approximation (e.g. 10 times of embedding dimension d or even more). This makes the computation cost still expensive for high dimensional data. To handle this issue, we adopt Structured Orthogonal Random Features (SORF), a variant of RFF for Gaussian kernel (Yu et al., 2016). Concretely, SORF replaces the linear random transformation matrix $\mathbf{W}$ with a structured orthogonal one:

$$\mathbf{W}_{\text{SORF}}{}^{1} = \frac{\sqrt{d}}{\sqrt{\tau}}\mathbf{H}\mathbf{D_1}\mathbf{H}\mathbf{D_2}\mathbf{H}\mathbf{D_3}, \tag{12}$$

where $D_i \in \mathbb{R}^{d\times d}, i = 1,2,3$, are diagonal "sign-flipping" matrices, with each diagonal entry sampled from the Rademacher distribution, and $\mathbf{H}$ is the normalized Walsh-Hadamard matrix.

While SORF is a biased estimator, its bias is negligible as long as d is not very small (Yu et al., 2016). Also, SORF induces much smaller variance than RFF when the projection dimension is fixed, which indicates that a prohibitively large D is not required to ensure the approximation accuracy. The most stand-out merit of SORF is that it can be computed in $\mathcal{O}(ND\log d)$ time by using fast Hadamard transformation (Fino & Algazi, 1976), nearly not requiring extra memory cost by using in-place operations. This endows our method with desirable scalability to larger dataset sizes and embedding dimensions. We term our method with RFF and SORF as *ESCo-RFF* and *ESCo-SORF* respectively, and evaluate them empirically in Sec. 5.

[1]The second dimension of $\mathbf{W}_{\text{SORF}}$ in Eq. 12 is restricted to d but could be extended to any dimension by concatenating multiple matrixes and removing redundant columns.

4.3 Relationships and Comparisons with Related Works

InfoNCE-based models. As pointed out in Sec. 4.1, the vanilla InfoNCE loss could be regarded as an extreme and special case of ESCo by adopting Gaussian kernel and specify the value of trade-off hyper-parameter, on top of which we further leverage Random Features to reduce the complexity. In this sense, ESCo could serve as a plug-in module that enhances other InfoNCE-based models, e.g. CMC (Tian et al., 2020a), MoCo (He et al., 2020), SimCLR (Chen et al., 2020) for images, SimCSE (Gao et al., 2021) for sentences, and GRACE (Zhu et al., 2020)/GraphCL (You et al., 2020) for nodes/graphs.

Asymmetric models. BYOL (Grill et al., 2020) and SimSiam (Chen & He, 2020) propose to use asymmetric structures like Stop-Gradient to avoid negative sampling. Despite their promising results and linear complexity (since no pair-wise distance is required), the rationales behind these approaches are still unclear[2]. Compared with them, ESCo can not only achieve linear scalability w.r.t. data sizes, its effectiveness for learning informative and redundancy-free representations is also theoretically grounded with the Multi-view Information Bottleneck principle.

Links to other works. To explain the learning behaviours of contrastive learning, a recent work (Wang & Isola, 2020) decomposes the InfoNCE objective into two terms, one for alignment and one for uniformity, and further proposes to optimize a weighted sum of them. Though we derive a similar objective form, our formulation takes a different perspective from the Multi-view Information Bottleneck and acts as a general framework that provides a unified view for existing contrastive approaches. Based on this, we dissect the effect of the trade-off hyper-parameter that plays a central role for guaranteeing effective learned representations. Moreover, another recent study (Tsai et al., 2021) combines several self-supervised objectives (e.g. contrastive learning and inverse predictive learning) to reach the MIB target. In comparison, the proposed ESCo does not require additional self-supervised objectives, and provides a more general implementation of the MIB principle.

5 Experiments

In this section, we evaluate ESCo empirically on various representation learning tasks. We select state-of-the-art contrastive methods on image and node representation learning as the baseline models and apply ESCo to boost them. We use `Pytorch` to implement all the baseline methods as well as the proposed two models. Experiments on images are conducted on four NVIDIA T4 GPUs, and other experiments are conducted on one NVIDIA T4 GPU. Specifically, we adopt `structure-net`[3] to do fast hadamard transformation, which enables much faster forward and backward computation of ESCo-SORF with CUDA accelerations.

5.1 Controlled experiments on Synthetic Datasets

To verify the effectiveness of the proposed model under ideal circumstances, we create a synthetic dataset, where the input data and the corresponding labels are ideally generated through the data generation process in Fig. 1(a). The flowchart of how we generate multiple views as well as labels is presented in Fig. 2, with detailed hyper-parameters presented in Appendix C. We select SimCLR (Chen et al., 2020) as the baseline model and the backbone of ESCo.

We randomly generate 4000 samples with 2048 input feature dimension. We follow the standard contrastive pretraining plus linear evaluation: we first train the model in an unsupervised manner with all the samples; at evaluation stage, we randomly split the dataset into 10 folds and use logistic regression with cross-validation to obtain the test accuracy. We adopt a 3-layer MLP as the encoder $f_\theta(\cdot)$, and set the temperature τ for all the methods as 1. Other hyper-parameters are all set identically for all the models and are listed detailedly in Appendix C.

Table 1: Test accuracy (%) on controlled synthetic data.

Method	Acc
SimCLR	38.11
ESCo-RFF ($\lambda = 1.6$)	40.52
ESCo-SORF ($\lambda = 1.5$)	40.86

[2] Tian et al. (2021) provide an analysis of their learning dynamics with two-layer models, which accounts for the reason why the two models do not fail with trivial solutions, yet it still remains as an open problem why they could learn informative representations.

[3] https://github.com/HazyResearch/structured-nets

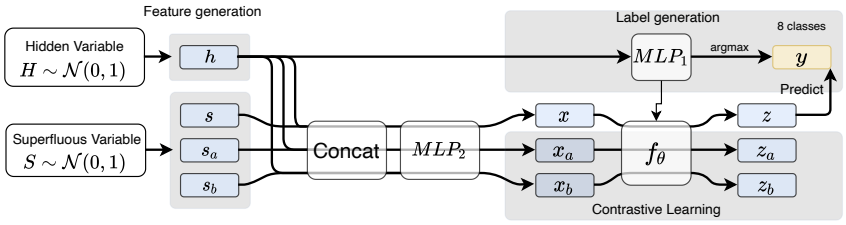

Figure 2: Data flow of synthetic dataset. The raw features are generated from two random variables: H is the hidden variable that decides the label, and S is the superfluous variable (e.g., controls augmentation-invariant features). The input feature $\boldsymbol{x}$ is the MLP output of the concatenation of hidden vector $\boldsymbol{h}$ sampled from H and superfluous vector $\boldsymbol{s}$ sampled from S. The input features of two views $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ have shared hidden vector but different superfluous vector with $\boldsymbol{x}$. f_θ is a learnable encoder parameterized by a 3-layer MLP. Other MLPs are randomly generated with fixed weights.

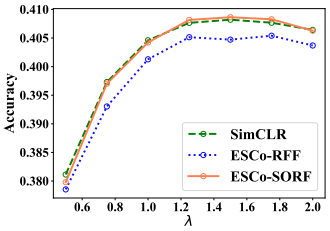

Figure 3: Effect of λ.

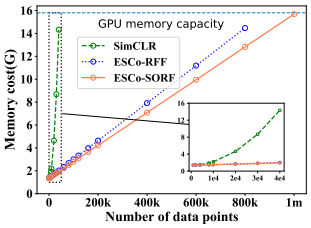

Figure 4: GPU memory cost.

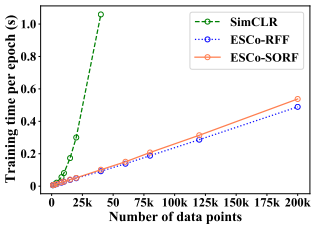

Figure 5: Time consumption.

In Table 1 we report the test accuracy of our methods compared with the baseline model SimCLR. ESCo-RFF ($\lambda = 1.5$) and ESCo-SORF ($\lambda = 1.6$) outperform SimCLR by 6.3% and 7.2% respectively. Note that SimCLR can be regarded as an extreme case of our method by setting $\lambda = 1/2\tau = 0.5$ and not using random features.

To investigate the effect of λ on the model performance, we gradually increase λ from 0.5 to 2.0 and study how the test accuracy changes. To study how the performance varies when using random features, we further equip SimCLR with the same trade-off hyper-parameter λ. Results are shown in Fig. 3. As we can see, SimCLR also benefits from the increase of λ (for vanilla SimCLR, $\lambda = 0.5$). Also, despite that ESCo-RFF suffers a bit performance drop, ESCo-SORF performs as well as SimCLR on prediction tasks. Another observation is that when λ becomes too large, the performance will begin to drop. This is reasonable as the second term in Eq. 9 gradually gets neglected with λ increasing, so it is important to select a proper λ that gives the best trade-off for better expressiveness and less redundancy.

We further validate the actual scalability performance of our methods against the baseline model. We gradually increase the number of data points from 1,000 to 1,000,000, and record the memory cost and execution time of our methods and SimCLR for computing the loss function. The results are shown in Fig. 4 and Fig. 5 respectively. Consistent with the theoretical complexity, the memory cost and time consumption of SimCLR grows quadratically w.r.t. the dataset sizes, and it runs out of memory of a 16G GPU when N exceeds 40,000. In comparison, ESCo-RFF and ESCo-SORF can be executed with linear memory cost, which makes our methods scalable to datasets with up to 1 million data points using a GPU with only 16G memory. The linear time complexity also enables them to be trained much faster on large-scale datasets.

5.2 Real-World Datasets

We then apply our methods on real-world unsupervised representation learning tasks, including image representations and node representations on graphs.

5.2.1 Image Representations

We evaluate our method on image representation learning tasks on `CIFAR-10`, `CIFAR-100` and `STL-10` datasets. Following previous practices (Chuang et al., 2020; Wang & Isola, 2020; Robinson et al., 2021), we choose SimCLR (Chen et al., 2020) as the base framework, and also for its simplicity

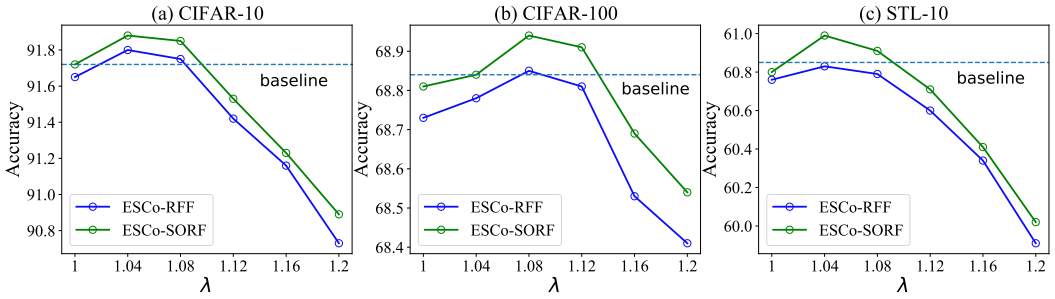

Figure 6: Top-1 accuracy for image classification on three public datasets. The dotted horizontal line represents the accuracy of baseline model (SimCLR).

Table 2: Accuracy, training time and memory cost on node classification tasks. ↑ indicates higher accuracy, and ↓ indicates lower training time / GPU memory cost.

Method	Cora (N: 2,708)			Citeseer (N: 3,327)			Pubmed (N: 19,717)		
	Acc	Time	Memory	Acc	Time	Memory	Acc	Time	Memory
GRACE	83.9	37.3s	2.6G	71.3	21.8s	3.1G	86.1	842.3s	14.5G
ESCo-RFF	84.3↑	22.2s↓	1.9G↓	71.6↑	14.8s↓	2.0G↓	86.1	203.9s↓	3.8G↓
ESCo-SORF	84.4↑	23.5s↓	2.0G↓	71.7↑	16.1s↓	2.1G↓	86.3↑	259.2s↓	4.1G↓

and strong performance, we use ResNet-50 (He et al., 2016) as the encoder (backbone). We adopt Adam optimizer (Kingma & Ba, 2014) with learning rate of 0.001. The temperature is set to $\tau = 0.5$ and the dimension of latent vectors is set to 128. All the models are trained with 500 epochs and evaluated using logistic regression. The results in Fig. 6 show that with a proper trade-off hyper-parameter λ, our method can boost the performance of the baseline model. Compared with the results on synthetic dataset in Sec. 5.1, we find that the optimal trade-off λ should be quite small, and when we increase it, the linear classification accuracy will drop obviously. The possible reasons include two aspects: 1) the redundant information between two views of images generated from random augmentation is not that considerable; 2) the multi-view redundancy assumption (Sridharan & Kakade, 2008) does not necessarily hold in real-world image classification datasets, which makes little room for benefits from Eq. 6.

5.2.2 Node Representations

We further evaluate our methods on unsupervised node classification tasks by adopting GRACE (Zhu et al., 2020) as the base framework and the competitor. GRACE is an InfoNCE-based contrastive learning method for node classification on graphs.

We conduct experiments on three commonly used citation networks: `Cora`, `Citeseer` and `Pubmed`. Each of them contains thousand-level nodes. We follow the setups as presented in (Zhu et al., 2020), and provide the detailed hyper-parameter settings in Appendix C. In order for fair comparisons, we adopt exactly the same hyper-parameters for the baseline model GRACE and our methods ESCo-RFF and ESCo-SORF, except the trade-off parameter λ. The results are presented in Table 2. As we can see, both ESCo-RFF and ESCo-SORF outperform the baseline model, with less training time and GPU memory cost. It is worth noting that on `Pubmed`, which contains almost $20,000$ nodes, our methods require much less training time and memory cost owing to their linear complexities.

6 Conclusion

We have proposed ESCo, a novel contrastive framework for unsupervised representation learning. ESCo is an instantiation of Information Bottleneck principle under multi-view self-supervised learning settings, and is proven to help learn informative and compressed representations. Compared with previous contrastive methods, our approach requires much less training time and GPU memory cost thanks to the adoption of Random Features, and consequently is scalable to large batch sizes. The experimental results on both image and graph data show its effectiveness for boosting the performance of existing contrastive learning approaches including SimCLR and GRACE.

REFERENCES

Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *ICML*, pp. 253–262, 2017.

Adrien Bardes, Jean Ponce, and Yann LeCun. Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pp. 531–540, 2018.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, Proceedings of Machine Learning Research, pp. 1597–1607, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *ICLR*, 2020.

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. In *NeurIPS*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NACAL*, pp. 4171–4186, 2019.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *ICLR*, 2020.

Bernard J. Fino and V. Ralph Algazi. Unified matrix treatment of the fast walsh-hadamard transform. *IEEE Transactions on Computers*, 25(11):1142–1146, 1976.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan 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 *NeurIPS*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pp. 9726–9735, 2020.

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 *ICLR*, 2019.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Quoc Le, Tamás Sarlós, Alex Smola, et al. Fastfood-approximating kernel expansions in loglinear time. In *ICML*, pp. 244–252, 2013.

Yazhe Li, Roman Pogodin, Danica J. Sutherland, and Arthur Gretton. Self-supervised learning with kernel dependence maximization. *arXiv preprint arXiv:2106.08320*, 2021.

Fanghui Liu, Xiaolin Huang, Yudong Chen, and Johan AK Suykens. Random features for kernel approximation: A survey on algorithms, theory, and beyond. *arXiv preprint arXiv:2004.11154*, 2020.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, pp. 69–84, 2016.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah A Smith, and Lingpeng Kong. Random feature attention. In *ICLR*, 2021.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *ICML*, pp. 5171–5180, 2019.

Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, pp. 1177–1884, 2007.

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

Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *ICLR*, 2021.

Karthik Sridharan and Sham M. Kakade. An information theoretic framework for multi-view learning. In *COLR*, pp. 403–414. Omnipress, 2008.

DJ Strouse and David J. Schwab. The deterministic information bottleneck. In *UAI*, 2016.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *ECCV*, pp. 776–794, 2020a.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? In *NeurIPS*, 2020b.

Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. In *ICML*, volume 139, pp. 10268–10278, 2021.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *ITW*, pp. 1–5, 2015.

Naftali Tishby, Fernando C. N. Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Yao-Hung Hubert Tsai, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. Self-supervised learning from a multi-view perspective. In *ICLR*, 2021.

Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *ICLR*, 2019.

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

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, pp. 9929–9939, 2020.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. In *NeurIPS*, 2020.

Felix Xinnan X Yu, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *NIPS*, pp. 1975–1983, 2016.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Hengrui Zhang, Qitian Wu, Junchi Yan, David Wipf, and Philip S Yu. From canonical correlation analysis to self-supervised graph neural networks. *arXiv preprint arXiv:2106.12484*, 2021.

Qinyi Zhang, Sarah Filippi, Arthur Gretton, and Dino Sejdinovic. Large-scale kernel methods for independence testing. *Statistics and Computing*, 28(1):113–130, 2018.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. Deep graph contrastive representation learning. *arXiv preprint arXiv:2006.04131*, 2020.

A ALGORITHMS

Algorithm 1: Pseudocode for ESCo-RFF and ESCo-SORF, PyTorch-like

```
def RFF_transform(input, w):
   # input: [N * d], input embedding matrix
   # w: [d * D], random projection matrix
   trans = torch.mm(input, w)
   D = w.size(1)
   output = torch.cat([torch.cos(trans), torch.sin(trans)], dim = 1) / np.sqrt(D)

   return output

# hadamard(input): function, fast hadamard transformation. input: N * d matrix.
    Complexity: O(N*d*logd)
def SORF_transform(input, D_m, T, temp:
   # input: [N * d], input embedding matrix
   # D_m: [T, 3, d], diagonal sign-flipping d * d
   matrixes, only containing diagonal terms.
   # T: int, T = D/d
   # temp: temperature hyper-parameter
   d = input.size(1)
   W = []
   for i in range(T):
      x = hadamard(input)
      x = hadamard(D_m[i][0] * x)
      x = hadamard(D_m[i][1] * x)
      x = D_m[i][2] * x * np.sqrt(d) / np.sqrt(temp)
      W.append(x)
   W = torch.cat(W, dim = 1)
   output = torch.cat([torch.cos(W), torch.sin(W)], dim=1) / np.sqrt(T*d)

   return output

def ESCo-loss(z1, z2, temp, lambda, D, method):
   z1 = normalize(z1, dim = 1)
   z2 = normalize(z2, dim = 1)
   loss_mse = (z1 - z2).pow(2).sum(1) * lambda # O(Nd)

   d = z1.size(1)
   if method == 'RFF':
      w = torch.randn(d,D) / np.sqrt(temp)
      tran1 = RFF_transform(z1, w) # O(NdD)
      tran2 = RFF_transform(z2, w) # O(NdD)
   elif method == 'SORF':
      T = int(D/d)
      D_m = torch.randint(0, 2, (T, 3, embedding.shape[1])).float()
      D_m = 2 * D_m - 1
      tran1 = SORF_transform(z1, D_m, T, temp) # O(NDlogd)
      tran2 = SORF_transform(z2, D_m, T, temp) # O(NDlogd)

   kernel_sum = torch.sum(tran2, dim = 0) # O(ND)
   loss_kernel = torch.log(torch.sum(tran1 * kernel_sim, dim = 1)) # O(ND)

   loss = (loss_mse + loss_kernel).mean()

   # Complexity of ESCo-RFF: O(NdD)
   # Complexity of ESCo-SORF: O(NDlogd)
```

B DATASETS AND EVALUATION PROTOCOLS

B.1 SYNTHETIC DATASET

The data generation process of the synthetic dataset has been briefly presented in Fig. 2. Here we introduce each part of the generation process detailedly.

First of all, we assume that both the hidden variable H and superfluous variable S are 64-dimensional vectors coming from a standard multivariate Gaussian distribution. For each data point, we generate $\boldsymbol{h}, \boldsymbol{s}, \boldsymbol{s}^A, \boldsymbol{s}^B, \in \mathbb{R}^{64}$, with each entry sampled from a standard normal distribution $\mathcal{N}(0, 1)$. Then the input hidden vectors for the data point itself, view A and view B are $[\boldsymbol{h}, \boldsymbol{s}]$, $[\boldsymbol{h}, \boldsymbol{s}^A]$ and $[\boldsymbol{h}, \boldsymbol{s}^B] \in \mathbb{R}^{128}$ respectively. These vectors are subsequently put into a 3-layer MLP (hidden dimension $128 - 256 - 256 - 1024$) with randomly initialized weights and softmax activation to generate the

Table 3: Statistics of node classification benchmarks

Dataset	#Nodes	#Edges	#Classes	#Features
Cora	2,708	10,556	7	1,433
Citeseer	3,327	9,228	6	3,703
Pubmed	19,717	88,651	3	500

input features $\boldsymbol{x}$, $\boldsymbol{x}^A$ and $\boldsymbol{x}^B$. h is then put into another 3-layer MLP (hidden dimension $64-256-8$) to generate its label (argmax is used to generate one-hot label).

For the encoder $f(\cdot)$, we adopt another 3-layer MLP (hidden dimension $1024-1024-1024-512$) with ReLU activation. The additional projector is not adopted here so the embedding got through the encoder is directly used to calculate the loss after l_2 normalization.

For evaluation, we generate the dataset containing 2048 data triples $\{(\boldsymbol{x}_i, \boldsymbol{x}_i^A, \boldsymbol{x}_i^B)\}_{i=1}^{2048}$, and the corresponding labels $\{\boldsymbol{y}_i\}_{i=1}^{2048}$. For training the encoder $f(\cdot)$, only the two views $\boldsymbol{x}^A$ and $\boldsymbol{x}^B$ are used. After the training process ends, the parameters of $f(\cdot)$ are frozen, and we use it to get the embeddings of $\boldsymbol{x}$, which is subsequently evaluated using logistic regression.

B.2 Image classification

`CIFAR-10`, `CIFAR-100` and `STL-10` are popular benchmarks for evaluating self-supervised models on image representation learning tasks with appropriate computation costs. Following previous practices (Chuang et al., 2020; Robinson et al., 2021) and for fair comparison with the baseline model, we adopt exactly the same hyper-parameters and image augmentation techniques for both SimCLR and ESCo (See Appendix C). For all the datasets, each method is trained for 500 epochs. Once the training ends, we evaluate the learned embeddings using logistic regression. The top-1 classification accuracy in Fig. 6.

B.3 Node classification

We evaluate our models on three node classification datasets: `Cora`, `Citeseer` and `Pubmed`. Each dataset is a citation network, where nodes denote papers and edges denote citation relationships. We provide the statistics of the three datasets in Table 3. For evaluation, we follow the practice in GRACE (Zhu et al., 2020) that randomly selects 10% nodes for training, 10% for validation and the remaining for testing.

We use the same data augmentation methods proposed in GRACE (Zhu et al., 2020): node feature masking and edge dropping. Following Zhu et al. (2020), we adopt a two-layer GCN (Kipf & Welling, 2017) as the encoder, and a two-layer MLP as the projector. At training stage, the full graphs generated through random augmentation are used to learn the GCN model, where all the other nodes within the graph are negative samples. At test stage, the original graph is put into the GCN model to get node representations, which are subsequently evaluated using logistic regression.

C Hyper-parameters

For the synthetic dataset, the hyper-parameters are set as:

- optimizer: Adam.
- training epochs: 1500.
- learning rate: 1e-5.
- weight decay: 0.
- embedding dimension: 512.
- random feature dimension: 2048.
- temperature τ: 1.0.

Table 4: Hyperparameters for node classification datasets. lr: learning rate; wd: weight decay; d_e: embedding dimension; d_p: projector dimension; $p_{e,i}$: edge dropping ratio of view i, $p_{f,2}$: feature masking ratio of view i; τ: temperature; λ_{rff} trade-off for RFF; λ_{sorf}: trade-off for SORF.

Dataset	Epochs	lr	wd	d_e	d_p	D_{rff}	$p_{e,1}$	$p_{e,2}$	$p_{f,1}$	$p_{f,2}$	τ	λ_{rff}	λ_{sorf}
Cora	400	1e-4	1e-5	512	512	1024	0.2	0.3	0.3	0.3	0.5	1.3	1.2
Citeseer	200	1e-3	1e-5	256	256	1024	0.2	0.0	0.3	0.2	0.9	2.0	2.1
Pubmed	1500	1e-4	1e-5	256	256	1024	0.4	0.1	0.0	0.2	0.7	1.5	1.5

- trade-off hyper-parameter λ: 1.6 for ESCo-RFF, 1.5 for ESCo-SORF.

For image classification on `CIFAR-10, CIFAR-100, STL-10`, the hyper-parameters are set as:

- optimizer: Adam.
- batch size: 512.
- training epochs: 500.
- learning rate: 1e-3.
- weight deacy: 0.
- encoder: ResNet-50.
- projector dimension: 512.
- random feature dimension: 1024.
- trade-off hyper-parameter λ for `CIFAR-10`: 1.04.
- trade-off hyper-parameter λ for `CIFAR-100`: 1.08.
- trade-off hyper-parameter λ for `STL-10`: 1.04.

For node classification on `Cora, Citeseer, Pubmed`, we list the hyper-parameters in Table.4.