Firstly, we would like to highlight the differences between BCL and previous work. BCL introduces analytical expressions for the class conditional probability densities of positive and negative examples based on the performance of the encoder, which is parameterized by macro-AUC metric α. This approach is different from previous studies and provides a more comprehensive understanding of the distribution and generation process of unlabeled data. In this work, we formalize the two crucial tasks of self-supervised contrastive learning, i.e., debiasing false negatives and mining hard negatives, in a Bayesian setting. These tasks are implicitly performed by jointly utilizing sample information (computable empirical distribution functions) and prior class information τ, which offers a flexible and principled self-supervised contrastive learning framework that avoids the high computational costs associated with explicit negative sampling. We would like to express our gratitude to the anonymous reviewers for recognizing the value of BCL approach and providing valuable suggestions to enhance the quality of this paper.

Secondly, we would like to highlight the innovative use of numerical experiments in our study. BCL simulates the generation process of unlabeled samples and presents a comprehensive set of numerical experimental results. ecause the numerical experiment does not contain noise and bias like a real dataset, it provides a better representation of how different estimators would behave in an ideal situation. While we did not include the performance of BCL on the ImageNet100 dataset in this version due to time constraints, we will incorporate it in the resubmitted version.

Finally, and most importantly, we would like to highlight the scalability of BCL for mining hard negatives without any parametric assumptions, as compared to HCL. The core idea is to use the class conditional probability density of true negatives to perform total probability decomposition and obtain the hard component as a sampling target, without relying on any parametric assumptions. By adopting this parameterization-free approach, we aim to reinforce the theoretical foundation of BCL and earn your continued endorsement for BCL approach.

We first present the overview of densities. Then we demonstrate the implementation process, experimental results, and

Table 2. Overview of densities.

Densities	Meaning	Description	Parameteric Assumption
$\phi(\hat{x})$	Anchor specific density of scores $p(\hat{x}; x, f)$.	Determined by the anchor x and encoder f .	No
$\phi_{(1)}(\hat{x})$	Density of order statistics $X_{(1)}$	Determined by $\phi(\hat{x})$.	No
$\phi_{(2)}(\hat{x})$	Density of order statistics $X_{(2)}$	Determined by $\phi(\hat{x})$.	No
$\phi_{\text{TN}}(\hat{x})$	Class conditional density of true negative $p(\hat{x}\|x^- \in \text{TN})$	Determined by α and order statistics $\phi_{(1)}(\hat{x})$, $\phi_{(2)}(\hat{x})$	No
$\phi_{\text{FN}}(\hat{x})$	Class conditional density of false negative $p(\hat{x}\|x^- \in \text{FN})$	Determined by α and order statistics $\phi_{(1)}(\hat{x})$, $\phi_{(2)}(\hat{x})$	No
$\psi(\hat{x})$	Class conditional density of hard true negatives $p(\hat{x}\|x^- \in \text{TN}, x^- \in \text{HARD})$	A component of ϕ_{TN} conditioned on a specific hardness level β.	No
$\phi_{\text{UN}}(\hat{x})$	Density of unlabeled data $\phi_{\text{UN}} = \tau^-\phi_{\text{TN}} + \tau^+\phi_{\text{FN}}$	Determined by class prior τ and class conditional densities.	No

implementation code of hard negative mining without any parameterization assumptions of BCL. Note that the class conditional density of true negatives is

$$\phi_{\text{TN}}(\hat{x}) \quad = \quad \alpha\phi_{(1)}(\hat{x}) + (1-\alpha)\phi_{(2)}(\hat{x})$$

where the first term $\alpha\phi_{(1)}(\hat{x})$ is the *easy negative sample component* that been correctly classified by the classifier (as shown in Fig. 2) , and the second term $(1-\alpha)\phi_{(2)}(\hat{x})$ is the *hard negative sample component* that been incorrectly classified by the classifier.

To avoid parametric assumptions, we decompose $\phi_{\text{TN}}(\hat{x})$ and select the component that corresponds to the hard negative sample as the target sampling distribution. To achieve this, we introduce a parameter $\beta \in [0, 1]$ to decompose the class conditional density of true negatives $\phi_{\text{TN}}(\hat{x})$ as follows:

$$\begin{aligned}
\phi_{\text{TN}}(\hat{x}) &= (1-\beta+\beta)[\alpha\phi_{(1)}(\hat{x}) + (1-\alpha)\phi_{(2)}(\hat{x})] \\
&= (1-\beta)\alpha\phi_{(1)}(\hat{x}) + \beta(1-\alpha)\phi_{(2)}(\hat{x}) \qquad (37) \\
&\quad + \beta\alpha\phi_{(1)}(\hat{x}) + (1-\beta)(1-\alpha)\phi_{(2)}(\hat{x}) \qquad (38)
\end{aligned}$$

In equation (37), the parameter $1-\beta$ controls the proportion of *easy negative sample components* $\alpha\phi_{(1)}(\hat{x})$, while β controls the proportion of *hard negative sample components* $(1-\alpha)\phi_{(2)}(\hat{x})$ that have been incorrectly classified by the classifier. Thus, Equation (37) can be interpreted as the density of hard true negative samples $p(\hat{x}, \text{HARD}|\text{TN})$ with a hardness level of β, a larger value of β (approaching 1) leads to $p(\hat{x}, \text{HARD}|\text{TN})$ contains higher proportion of *hard negative sample component* that have been incorrectly classified by the classifier.

Similarly, Equation (38), the mirrored counterpart of Equation (37), represents the density of easy true negative samples $p(\hat{x}, \textsc{Easy}|\textsc{Tn})$ with an easiness level of β. A larger value of β (approaching 1) leads to Equation (38) contains higher proportion of *easy negative sample component* that have been correctly classified by the classifier.

The above algebraic transformation can be viewed as total probability decomposition of the class conditional distribution $\phi_{\textsc{Tn}}(\hat{x})$:

$$\begin{aligned}
\phi_{\textsc{Tn}}(\hat{x}) &\triangleq p(\hat{x}|\textsc{Tn}) \\
&= p(\hat{x}, \textsc{Hard} = \beta|\textsc{Tn}) + p(\hat{x}, \textsc{Easy} = \beta|\textsc{Tn}) \\
&= (1-\beta)\alpha\phi_{(1)}(\hat{x}) + \beta(1-\alpha)\phi_{(2)}(\hat{x}) + \ \beta\alpha\phi_{(1)}(\hat{x}) + (1-\beta)(1-\alpha)\phi_{(2)}(\hat{x})
\end{aligned}$$

The distribution $p(\hat{x}, \textsc{Hard}|\textsc{Tn})$ in Eq (37) is a newly defined distribution measured on a separate sample space consisting of hard negative examples, which is a subset of population of true negative examples. It is worth noting that $p(\hat{x}, \textsc{Hard}|\textsc{Tn})$ is already conditioned on the true principle since it is a component of $\phi_{\textsc{Tn}}(\hat{x})$. Additionally, the hardness level β is conditioned on the hard principle since it controls the proportion of the *hard negative sample component* that has been incorrectly classified by the classifier. So the desired sampling distribution for drawing $\{x_i^-\}_{i=1}^N$ conditioning on both true principle and hard principle can be derived as:

$$\begin{aligned}
\psi(\hat{x}; \alpha, \beta) &\triangleq p(\hat{x}|\text{TN}, \textsc{Hard}) \\
&= p(\hat{x}, \textsc{Hard} = \beta|\text{TN})/p(\textsc{Hard} = \beta|\text{TN}) \qquad (39)
\end{aligned}$$

$p(\textsc{Hard} = \beta|\text{TN})$ is the normalization constant, which can be calculated by the marginal integration of $p(\hat{x}, \textsc{Hard} = \beta|\textsc{Tn})$ over $\hat{x}$

$$\begin{aligned}
p(\textsc{Hard} = \beta|\text{TN}) &= \int_{-\infty}^{\infty} p(\hat{x}, \textsc{Hard} = \beta|\textsc{Tn})d\hat{x} \\
&= \int_{-\infty}^{\infty} (1-\beta)\alpha\phi_{(1)}(\hat{x}) + \beta(1-\alpha)\phi_{(2)}(\hat{x})d\hat{x} \\
&= (1-\beta)\alpha + \beta(1-\alpha) \qquad (40)
\end{aligned}$$

Finally, the desired sampling distribution for drawing $\{x_i^-\}_{i=1}^N$ conditioning on both true principle and hard principle given by Eq (39) can be calculated as:

$$\psi(\hat{x}; \alpha, \beta) = \frac{(1-\beta)\alpha\phi_{(1)}(\hat{x}) + \beta(1-\alpha)\phi_{(2)}(\hat{x})}{(1-\beta)\alpha + \beta(1-\alpha)} \qquad (41)$$

As a point of transition between hardness and easiness, when $\beta = 0.5$,

$$\psi(\hat{x}; \alpha, \beta) = \phi_{\textsc{Tn}}(\hat{x}),$$

which indicates that samples are drawn from the original class conditional density of true negatives. On the other hand, when $\beta = 1$, $\psi(\hat{x}; \alpha, \beta) = \phi_{(2)}(\hat{x})$, the target of sampling distribution only contains hard negative samples that have been misclassified by the classifier.

It's worth noting that since we have not introduced the un-normalized von Mises-Fisher (vMF) assumption, the distribution $\psi(\hat{x}; \alpha, \beta)$ is now a normalized distribution, and we can calculate its normalization constant $p(\textsc{Hard} = \beta|\text{TN})$ exactly using Eq (40).

Now we have in batch N i.i.d unlabeled samples $\{\hat{x}_i\}_{i=1}^N \sim \phi_{\text{UN}} = \tau^-\phi_{\text{TN}} + \tau^+\phi_{\text{FN}}$. In addition, we have a desired sampling distribution, denoted by ψ, of hard true negative samples, we can approximate the expectation scores over hard and true samples using classic Monte-Carlo importance sampling (Hesterberg, 1988; Bengio & Senécal, 2008):

$$\begin{aligned}
\mathbb{E}_{\hat{x}\sim\psi}\hat{x} &= \int_{+\infty}^{+\infty} \hat{x}\frac{\psi(\hat{x}; \alpha, \beta)}{\phi_{\text{UN}}(\hat{x})}\phi_{\text{UN}}(\hat{x})d\hat{x} \\
&= \mathbb{E}_{\hat{x}\sim\phi_{\text{UN}}}\hat{x}\frac{\psi(\hat{x}; \alpha, \beta)}{\phi_{\text{UN}}(\hat{x})} \\
&\simeq \frac{1}{N}\sum_{i=1}^{N}\omega_i\hat{x}_i \qquad (42)
\end{aligned}$$

where ω_i is the density ratio between desired sampling distribution ψ and unlabeled data distribution ϕ, which can be calculated by:

$$\omega_i(\hat{x}_i; \alpha, \beta) \quad = \quad \frac{\psi(\hat{x}; \alpha, \beta)}{\phi_{\text{UN}}(\hat{x})} \tag{43}$$

$\omega_i(\hat{x}_i; \alpha, \beta)$ is a function of empirical cumulative distribution $\Phi_n(\hat{x})$ (sample information) and class prior probability τ (prior information). Since the desired sampling distribution $\psi(\hat{x}; \alpha, \beta)$ now is normalized, the revised version of BCL do not involves the calculation of the partition function any more. The revised BCL therefore involves only two steps:

```
step-1: calculating the empirical C.D.F.
```

```
step-2: calculating the weights by Eq(43)
```

The implementation code of BCL without parametric assumption is available at the anonymous repository:

`https://anonymous.4open.science/r/BCL/main.py`

The importance weight $\omega(\hat{x}, \alpha, \beta)$ essentially assigns customized weights to the N unlabeled samples.

The parameter α corresponds to the encoder's macro-AUC metric for false negative debiasing, controlling the distribution of unlabeled scores $\phi_{\text{UN}}(\hat{x})$ (better encoder encodes dissimilar data points with different class labels more orthogonal), which can be empirically estimated using a validation dataset during the training process.

The parameter β corresponds to the proportion of misclassified *hard negative component* in the desired sampling distribution $\psi(\hat{x}; \alpha, \beta)$, which controls the hardness level required for specific task scenarios. Since Eq (37) is a component of the total probability decomposition of $\phi_{\text{TN}}(\hat{x})$, varying the value of β actually changes the separate sample space consists of negative samples with different hardness level, without altering the asymptotically unbiased estimation of Eq (42) for the scores of true negatives.

Experimental details:

(1) α represents the macro AUC of the encoder, which is fixed to a function that grows at a constant rate with the training epoch. Specifically, we set α as follows: $\alpha = 0.5 + 0.35/400 * epoch$, where the training epoch is fixed at 400 epochs.

(2) β is the desired hardness level on hard negative samples, and we set it as $\beta = 1$.

The experimental setup is identical to that of DCL, HCL, and SimCLR. We fix the number of negative samples N as 510 and conduct the experiment six times to obtain six results of ACC1: [92.24, 92.26, 92.19, 92.23, 92.27, 92.25, 92.26]. At the 5% significance level, BCL outperformed HCL on the CIFAR10 dataset. To replicate the BCL and comparative learning methods, please use the following command.

```
python main.py --dataset_name cifar10 --batch_size 256 --estimator BCL
python main.py --dataset_name cifar10 --batch_size 256 --estimator HCL
python main.py --dataset_name cifar10 --batch_size 256 --estimator DCL
python main.py --dataset_name cifar10 --batch_size 256 --estimator SimCLR
```