**Self-Contrastive Learning: An Efficient Supervised Contrastive Framework with Single-view and Sub-network**

**Sangmin Bae** 1  
**Sungnyun Kim** 1  
**Jongwoo Ko** 1  
**Gihun Lee** 1  
**Seungjong Noh** 2  
**Se-Young Yun** 1

**Abstract**

This paper proposes an efficient supervised contrastive learning framework, called **Self-Contrastive (SelfCon) learning**, that self-contrasts within multiple outputs from the different levels of a multi-exit network. SelfCon learning with a single-view does not require additional augmented samples, which resolves the concerns of multi-viewed batch (e.g., high computational cost and generalization error). Unlike the previous works based on the mutual information (MI) between the multi-views in unsupervised learning, we prove the MI bound for SelfCon loss in a supervised and single-viewed framework. We also empirically analyze that the success of SelfCon learning is related to the regularization effect from the single-view and sub-network. For ImageNet, SelfCon with a single-viewed batch improves accuracy by +0.3% with 67% memory and 45% time of Supervised Contrastive (SupCon) learning, and a simple ensemble of multi-exit outputs boost performance up to +1.4%.

1. **Introduction**

Recent studies have examined the success of deep neural networks by investigating how neural networks can encode representations with rich information (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017; Hjelm et al., 2018; Saxe et al., 2019a). Among the various approaches suggested, contrastive loss functions, which are designed to maximize the lower bound of mutual information (MI) between the target and the context, have achieved considerable success in self-supervised representation learning first (Gutmann & Hyvärinen, 2010; Oord et al., 2018) and supervised learning recently (Khosla et al., 2020; Gunel et al., 2020; Wang et al., 2021). The main objective of the contrastive loss functions in supervision is to make representations from the same class closer and representations from different classes farther. To this end, Khosla et al. (2020) define positive samples, i.e., augmented samples from the same image or (augmented) images sharing the same class label, and negative samples, i.e., all other samples, for every data batch.

Contrasting two random augmented samples, which is often referred to as a multi-viewed batch, has shown impressive results in representation learning (Chen et al., 2020; He et al., 2020; Grill et al., 2020; Caron et al., 2020; Chen & He, 2020; Khosla et al., 2020), yet a multi-viewed batch causes the following issues. (1) A multi-viewed batch doubles the batch size, which places an enormous burden on memory and computation. (2) Oracle needs to choose its augmentation policies carefully (Chen et al., 2020; Tian et al., 2020; Caron et al., 2020; Kim et al., 2020). (3) A multi-viewed contrastive task is domain-specific because data-level augmentation, such as image cropping and color jittering, requires specific domain knowledge (Verma et al., 2021). In spite of these concerns, a supervised contrastive learning framework (Khosla et al., 2020) still relies on multi-views, although they are not necessarily needed when usable label information is available.

In this work, we propose a novel contrastive learning framework for supervision, called **Self-Contrastive (SelfCon) learning**, that does not require additional augmented samples. Instead, SelfCon learning uses a multi-exit framework (Teerapittayanon et al., 2016; Zhang et al., 2019a;b) where sub-networks produce multiple outputs for the same input, and self-contrasts within multiple outputs from the different levels of a single network. For training, SelfCon learning can use a multi-viewed batch as well. However, the multi-exit framework already generates positive pairs from a single image, which can replace an augmentation-based multi-view. In Figure 1, we compare SelfCon learning and Supervised Contrastive (SupCon, Khosla et al. (2020)) learning.

SelfCon learning achieves comparable or better performance than SupCon with greatly reduced memory and time cost. SelfCon learning improves the classification performance of the encoder network (1) because the intermediate layers learn the class-related information in the last fea-
We summarize the contributions of the paper as follows:

As the cost of data labeling increases exponentially, there
from the previous methods based on the MI between the
multi-views in unsupervised learning (Bachman et al., 2019;
Tian et al., 2019a), we prove the MI bound in a supervised
and single-viewed framework. Second, we empirically show
that SelfCon reduces the generalization error with the single-
view and sub-network. The former prevents the encoder
from overfitting to each instance, and the latter regularizes
the intermediate feature to be similar to the last feature.

We summarize the contributions of the paper as follows:

[S3] We propose Self-Contrastive learning, an efficient sup-
ervised contrastive framework between multiple features
from different levels of a single network.

[S4] We guarantee that SelfCon loss is the lower bound of
label-conditional MI between the intermediate and the last
features. Then, we discuss the theoretical evidence of an
improved classification performance from this MI bound.

[S5.1, S5.2] SelfCon learning efficiently achieves higher
classification accuracy for various benchmarks than do
cross-entropy and SupCon loss. Our empirical study of
MI estimation provides evidence for superior performance.

[S5.3, S5.4] We investigate the advantages of the single-
viewed batch in terms of the computation cost and the gen-
eralization error. We also identify that SelfCon learning
benefits from the sub-network owing to the regularization
effect, ensemble prediction, and vanishing gradient.

2. Related Works

2.1. Contrastive Learning

As the cost of data labeling increases exponentially, there
is an increased need for acquiring representation in unsu-
pervised scenarios. After Oord et al. (2018) proposed an
InfoNCE loss (also called a contrastive loss), contrastive
learning-based algorithms began to show a remarkable im-
provement in performance (Chen et al., 2020; He et al.,
2020; Grill et al., 2020; Caron et al., 2020; Chen & He,
2020). Most previous works used a Siamese network, which
is a weight-sharing network applied on two augmented
inputs, with negative pairs (SimCLR (Chen et al., 2020),
MoCo (He et al., 2020)), momentum encoders (MoCo (He
et al., 2020), BYOL (Grill et al., 2020)), online clustering
(SwAV (Caron et al., 2020)), or a stop-gradient oper-
ation (SimSiam (Chen & He, 2020)). While the softmax
form is frequently used for the contrastive loss (SimCLR,
MoCo), recent state-of-the-art algorithms utilize an MSE
loss (BYOL, SimSiam), or a cross-entropy loss (SwAV,
DINO (Caron et al., 2021)). Khosla et al. (2020), inspired
by success in self-supervised learning, proposed a label-
based contrastive loss in supervision, named SupCon loss.
SupCon learning has also been extended to semantic seg-
mentation (Wang et al., 2021) and language tasks (Gunel
et al., 2020). While SupCon loss utilizes the output fea-
tures from two random augmentations, we also contrast the
features from different network paths by introducing the
multi-exit framework (Teerapittayanon et al., 2016; Zhang
et al., 2019a;b). We further propose a novel loss function
that we can apply on a single-viewed batch.

2.2. Mutual Information

MI is a measure for quantifying the information held in
a random variable about another variable, and it has been
used as a powerful tool to open the black box of deep neu-
networks (Tishby & Zaslavsky, 2015; Shwartz-Ziv &
Tishby, 2017; Sa et al., 2019a). As it is difficult to com-
pute the MI exactly (Paninski, 2003), several works have
proposed variational MI estimators based on neural net-
works, e.g., InfoNCE (Oord et al., 2018), MINE (Belghazi
et al., 2018), NWJ (Nguyen et al., 2010), ML-CPC (Song
& Ermon, 2020), and SMILE (Song & Ermon, 2019). Re-
cently, MI estimator-based objectives have been proposed
to improve the performance of contrastive learning (DIM
(Hjelm et al., 2018), AMDIM (Bachman et al., 2019), CMC
(Tian et al., 2019a)) and knowledge distillation (CRD (Tian
et al., 2019b), VID (Ahn et al., 2019)). Among previous
approaches, we have been highly motivated by those that
aim to increase the MI between the intermediate and the
last features. DIM and AMDIM propose novel contrastive
losses between the global features and local features (i.e.,
all pixels or patches of the intermediate features), whereas
we contrast the refined local features via sub-networks. VID
makes the student learn the distribution of the activations in
the auxiliary teacher’s intermediate features. Our method
differs from VID in that we self-distill within the network
itself. Zhang et al. (2019a;b) share an idea similar to that of
the aforementioned works, but they implicitly increase MI
using Kullback–Leibler (KL) divergence loss.

3. Self-Contrastive Learning

We propose a new supervised contrastive loss that max-
imizes the similarity of the outputs from different net-
work paths by introducing the multi-exit framework.
We define an encoder structure, using \( F \) as a backbone
network and \( G \) as a sub-network, that shares the backbone’s
parameters up to some intermediate layer. \( T \) denotes the
sharing layers that produce the intermediate feature. Note
that \( F \) and \( G \) include the projection head after the encoder.
We highlight the positive and negative pairs with respect to
an anchor sample, following Figure 1.
We aim to maximize the similarity between
SelfCon loss, which forms a self-
contrastive task for every output, including the features
from the sub-network.

\[ L_{self} = \sum_{i \in I} \left[ -\frac{1}{|P_i|} \sum_{p \in P_i} \omega(x_i)^\top \omega(x_p) + \log \left( \sum_{p \in P_i} e^{F(x_i)^\top F(x_p)} + \sum_{n \in N_i} e^{F(x_i)^\top F(x_n)} \right) \right] \]

where \( \Omega = \{F, G\} \) is a function set of the backbone network and the sub-network. \( \omega_1 \) is a function that generates positive pair, and \( \omega_2 \) is for generating every contrastive pair from a multi-exit network. We include an anchor sample to the positive set when the output feature is from a different exit path, i.e., \( P_{ij} \leftarrow P_i \cup \{i\} \) when \( \omega \neq \omega_j \). For example, \( G(x_i) \) is also an positive pair for \( F(x_i) \). We also omit \( \tau \) and the dividing constant. Whereas prevalent contrastive approaches (Khosla et al., 2020; Chen et al., 2020; He et al., 2020; Grill et al., 2020) force a multi-viewed batch generated by data augmentation, the sub-network in SelfCon learning plays a role as the augmentation and provides an alternative view on the feature space. Therefore, without the additional augmented samples, we formulate our SelfCon loss function with a single-viewed batch (SelfCon-S), as well as with a multi-viewed batch (SelfCon-M).

We can further use multiple sub-networks, i.e., \( \Omega = \{F, G_1, G_2, \ldots\} \). Appendix B.4 presents the classification performance of the expanded network, but there was no significant improvement from that of a single sub-network. Thus, we have efficiently used a single sub-network throughout our paper.
4. Discussions

In this Section, we discuss theoretical evidence for the success of SelfCon learning. We summarize the discussion as follows: SelfCon learning improves the classification performance by encouraging the intermediate feature to have more label information in the last feature.

Discussion 4.1. How does SelfCon loss encourage the intermediate feature to learn the label information in the last feature?

Generally, prior works (Oord et al., 2018; Hjelm et al., 2018) support the success of unsupervised contrastive learning from the connection to the MI. In this sense, in Proposition 4.1, we first prove the connection between a supervised contrastive loss and the MI of positive pairs. In Proposition 4.2, we then provide the MI bound within a single-viewed batch using the sub-network feature.

Proposition 4.1. Let \( \mathbf{x} \) and \( \mathbf{z} \) be different samples that share the same class label \( c \). Then, with some discriminator function modeled by a neural network \( \mathbf{F} \) and \( 2(K - 1) \) negative sample size, SupCon loss maximizes the lower bound of conditional MI between the output features of a positive pair.

\[
\log(2K-1) - \mathcal{L}_{\sup}(\mathbf{x}, \mathbf{z}; \mathbf{F}, K) \leq \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{F}(\mathbf{z})|c) \tag{3}
\]

Proposition 4.2. Denote \( \mathcal{L}_{\text{self}, s} \) as SelfCon loss with single-viewed batch. SelfCon-S loss maximizes the lower bound of MI between the output features from the backbone and the sub-network.

\[
\log(2K-1) - \mathcal{L}_{\text{self}, s}(\mathbf{x}; \{\mathbf{F}, \mathbf{G}\}, K) \leq \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{G}(\mathbf{x})|c) \tag{4}
\]

SupCon and SelfCon-S loss have a negative sample size of \( 2(K - 1) \) because of the augmented negative pairs for SupCon and the sub-network features for SelfCon-S. Note that SelfCon-M loss has an upper bound similar to Eq. 3 and Eq. 4 (refer to Appendix A.2).

We extend the above MI bound to the MI between the intermediate and last feature of a backbone. Although MI is ill-defined between the variables with deterministic mapping, previous works view the training of a neural network as a stochastic process (Shwartz-Ziv & Tishby, 2017; Goldfeld et al., 2019; Sax et al., 2019b). Thus, encoder features are considered as random variables, which allows us to define and analyze the MI between the features.

Proposition 4.3. As \( \mathbf{F}(\mathbf{x}) \) and \( \mathbf{G}(\mathbf{x}) \) are conditionally independent given the intermediate representation \( \mathbf{T}(\mathbf{x}) \), they formulate a Markov chain: \( \mathbf{G} \leftrightarrow \mathbf{T} \leftrightarrow \mathbf{F} \) (Cover, 1999). Then, the following is satisfied.

\[
\mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{G}(\mathbf{x})|c) \leq \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x})|c) \tag{5}
\]

Proposition 4.3 implies that the encoder makes better representation by the intermediate features that learn the class-related information from the last features. Although SelfCon loss does not explicitly include the term \( \mathbf{T}(\mathbf{x}) \), minimizing SelfCon loss is almost equivalent to increasing the information of \( \mathbf{T}(\mathbf{x}) \) because we usually use one fully-connected layer as the non-sharing layer of the sub-network \( \mathbf{G} \).

Discussion 4.2. How does increasing \( \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x})|c) \) improve classification performance?

To understand the information that SelfCon loss maximizes, we decompose the r.h.s. of Eq. 5 as follows:

\[
\mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x})|c) = \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x}), c) - \mathcal{I}(\mathbf{F}(\mathbf{x}); c) \tag{6}
\]

\[
\mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x})|c) \geq \mathcal{I}(\mathbf{F}(\mathbf{x}); \mathbf{T}(\mathbf{x})) + \mathcal{I}(\mathbf{F}(\mathbf{x}); c|\mathbf{T}(\mathbf{x})) - \mathcal{I}(\mathbf{F}(\mathbf{x}); c). \tag{7}
\]

(\( \square \)) implies that \( \mathbf{T}(\mathbf{x}) \) is distilled with refined information (not conditional with respect to \( c \)) from \( \mathbf{F}(\mathbf{x}) \), so the encoder can produce better representation (Hjelm et al., 2018; Ahn et al., 2019). On the other hand, (\( \blacksquare \)) is interaction information (Yeung, 1991) that measures the influence of \( \mathbf{T}(\mathbf{x}) \) on the amount of shared information between \( \mathbf{F}(\mathbf{x}) \) and \( c \). Increasing this interaction information means the intermediate feature enhances the correlation between the last feature and the label. Therefore, when we jointly optimize (\( \square + \blacksquare \)), the intermediate and last features have aligned label information.

In this sense, SelfCon loss is based on the InfoMax principle (Linsker, 1989), which is about learning to maximize the MI between the input and output of a neural network. It has been proved that InfoMax-based loss regularizes intermediate features and improves performance in semi-supervised (Rasmus et al., 2015) and knowledge transfer (Ahn et al., 2019) domains. Similar to the previous works, SelfCon loss increases the classification accuracy by regularizing the intermediate feature to have class-related information aligned with the last feature.

Discussion 4.3. Is SelfCon loss applicable to unsupervised representation learning?

The unsupervised version of SelfCon loss is a lower bound of (\( \square \)) in Eq. 6. By maximizing only (\( \square \)), the last feature may follow the intermediate feature, learning redundant information about the input.\(^1\) This could be the reason why SelfCon learning does not work in an unsupervised environment (refer to Appendix D.1). However, to mitigate this problem, we propose in Appendix D.2 a loss function to prevent the backbone from following the sub-network. For

\(^1\)In supervision, a suboptimal case where \( \mathbf{T}(\mathbf{x}) \) becomes a sink for \( \mathbf{F}(\mathbf{x}) \) does not happen because the deeper layers have a larger capacity for label information (Shwartz-Ziv & Tishby, 2017).
Table 1. The results of the linear evaluation for various datasets. In the supervised setting, we compare our SelfCon-M and SelfCon-S with cross-entropy (CE) loss, supervised contrastive loss with multi-view (SupCon), and with single-view (SupCon-S). Bold type is for all the values of which the standard deviation range overlaps with that of the best accuracy. We used the same batch size of 1024 and a learning rate of 0.5 as Khosla et al. (2020) did in CIFAR experiments.

| Method     | Single-View | ResNet-18       | ResNet-50       |
|------------|-------------|-----------------|-----------------|
|            |             | CIFAR-10 | CIFAR-100 | Tiny-ImageNet | CIFAR-10 | CIFAR-100 | Tiny-ImageNet |
| CE         | ✓           | 94.7 ±0.1 | 72.9 ±0.1 | 57.5 ±0.3 | 94.9 ±0.2 | 74.8 ±0.1 | 62.3 ±0.4 |
| SupCon     |             | 94.7 ±0.2 | 73.0 ±0.0 | 56.9 ±0.4 | 95.6 ±0.1 | 75.5 ±0.2 | 61.9 ±0.2 |
| SupCon-S   | ✓           | 94.9 ±0.0 | 73.9 ±0.1 | 58.4 ±0.3 | 95.8 ±0.1 | 76.7 ±0.1 | 62.0 ±0.2 |
| SelfCon-M (ours) | ✓ | 95.0 ±0.1 | 74.9 ±0.1 | 59.2 ±0.0 | 95.5 ±0.1 | 76.9 ±0.1 | 63.0 ±0.2 |
| SelfCon-S (ours) | ✓ | 95.3 ±0.2 | 75.4 ±0.1 | 59.8 ±0.4 | 95.7 ±0.2 | 78.5 ±0.3 | 63.7 ±0.2 |

*We have re-implemented SupCon method (Khosla et al., 2020) and also run their official code for credibility, but the accuracy was slightly lower than their reported numbers.

SupCon-S sets $I$ as $\{1, ..., B\}$ in Eq. 1. Although Khosla et al. (2020) did not propose the version of the single-view, we implemented SupCon-S since it is worth investigating the effect of multi-viewed batch.

5. Experiment

We present the image classification accuracy for standard benchmarks, such as CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), Tiny-ImageNet (Le & Yang, 2015), ImageNet-100 (Tian et al., 2019a), and ImageNet (Deng et al., 2009), and extensively analyze the results. We report the mean and standard deviation of top-1 accuracy over three random seeds. We also tuned the learning rate for the various benchmarks. Furthermore, we used the optimal structure and position of the sub-network for all the network architectures. The overall performance was comparable to or better than the baselines. The complete implementation details and hyperparameter tuning results are presented in Appendix B.

5.1. Representation Learning

We measured the classification accuracy of the representation learning protocol (Chen et al., 2020), which consists of 2-stage training: (1) pretraining an encoder network and (2) fine-tuning a linear classifier with the frozen encoder (called a linear evaluation).

Small-scale benchmark The classification accuracy is summarized in Table 1. Interestingly, the loss functions in the single-viewed batch, such as SupCon-S and SelfCon-S, outperform their multi-view counterparts in all settings. Furthermore, our SelfCon learning, which trains using the sub-network, shows higher classification accuracy than CE and SupCon. The effects of the sub-network are analyzed in Section 5.4.

Table 2. The classification accuracy on ResNet-18 and ResNet-50 for ImageNet-100. We summarized the ratio of memory (GiB / GPU) and Time (sec / step) based on those of SelfCon-S in each backbone.

| Method     | ResNet-18 | ResNet-50 |
|------------|-----------|-----------|
|            | Mem. | Time | Acc@1 | Mem. | Time | Acc@1 |
| CE         | -    | 83.7 | -     | -    | 86.4 | -     |
| SupCon     | ×1.5 | ×2.1 | 85.5  | ×1.7 | ×1.7 | 88.2 |
| SupCon-S   | ×1.0 | ×1.0 | 84.9  | ×0.9 | ×0.8 | 87.8 |
| SelfCon-M  | ×1.6 | ×2.1 | 85.8  | ×1.8 | ×2.2 | 88.9 |
| SelfCon-S  | ×1.0 | ×1.0 | 86.3  | ×1.0 | ×1.0 | 88.5 |

Table 3. The classification accuracy on ResNet-18 for ImageNet. The performance of CE is reported only in $B = 1024$. Every memory and time is reported by comparing to those of SelfCon-S in each batch size.

| Method     | $B = 1024$ | $B = 2048$ |
|------------|------------|------------|
|            | Mem. | Time | Acc@1 | Mem. | Time | Acc@1 |
| CE         | -    | 69.4 | -     | -    | 69.4 | -     |
| SupCon     | ×1.6 | ×2.2 | 70.9  | ×1.5 | ×2.2 | 71.2 |
| SupCon-S   | ×0.9 | ×1.0 | 69.2  | ×0.9 | ×1.0 | 70.2 |
| SelfCon-M  | ×1.7 | ×2.3 | 71.2  | ×1.7 | ×2.3 | 71.6 |
| SelfCon-S  | ×1.0 | ×1.0 | 70.3  | ×1.0 | ×1.0 | 71.5 |

Large-scale benchmark We first experimented on the ImageNet-100 benchmark, of which 100 classes were randomly sampled (Tian et al., 2019a), to verify that SelfCon learning has the same effect on large-scale datasets. Table 2 presents the consistent performance improvement of SelfCon learning. In particular, SelfCon-S shows a higher cost-to-accuracy ratio than SupCon-S and SupCon. That is, SelfCon-S achieves higher accuracy with a cost similar to SupCon-S or about half the cost of SupCon.

Table 3 summarizes the full-scale ImageNet results. Because SelfCon-M has a high computational cost, it consistently shows the best performance for every batch size.
Table 4. 1-stage training on ResNet architectures. † describes a modification to 1-stage training with a multi-exit framework. Parentheses indicate the sub-network’s accuracy. The last row is the results for 2-stage SelfCon-S.

| Method          | ResNet-18 CIFAR-100 | ResNet-18 ImageNet-100 | ResNet-50 CIFAR-100 | ResNet-50 ImageNet-100 |
|-----------------|---------------------|------------------------|---------------------|------------------------|
| CE              | 72.9                | 83.7                   | 74.8                | 86.4                   |
| CE w/ Sub†      | 73.5 (70.2)         | 84.0 (83.1)            | 76.2 (72.3)         | 86.7 (85.5)            |
| SD†             | 73.7 (71.5)         | 84.7 (83.2)            | 76.1 (73.3)         | 86.7 (85.9)            |
| SelfCon-S†      | 74.5 (70.4)         | 84.8 (84.4)            | 76.8 (72.6)         | 87.3 (85.6)            |
| SelfCon-S       | 75.4 (70.1)         | 86.3 (85.0)            | 78.5 (72.3)         | 88.5 (87.5)            |

However, SelfCon-S is comparable to SelfCon-M and even superior to SupCon despite the greatly reduced cost. The poor performance of SupCon-S, which consumes an amount of memory and time similar to SelfCon-S, reflects the superiority of the SelfCon framework for a single-viewed batch.

1-stage training To compare SelfCon-S with the standard supervised training with sub-network, we experimented with a 1-stage training framework, i.e., not decoupling the encoder pretrained and linear evaluation. With a multi-exit framework, Zhang et al. (2019a) proposed Self-Distillation (SD), which distills logit information within the network itself, i.e., \( \mathcal{L}_\text{SD} = \alpha \mathcal{L}_\text{CE} + (1 - \alpha) \mathcal{L}_\text{KL} \). We replaced the KL divergence term with \( \mathcal{L}_{\omega \leftarrow \mathcal{L}} \). From Table 4, we observed that simply adding cross-entropy loss to the sub-network (CE w/ Sub) improved the backbone network’s classification performance. However, the results of SD suggest a saturation of the backbone’s accuracy even when the classifier of the sub-network converges well. We would like to highlight that the SelfCon loss in 2-stage training still demonstrated the best classification accuracy, as Khosla et al. (2020) suggested that representation learning mitigates the poor generalization performance of CE-based 1-stage training.

5.2. Mutual Information Estimation

We argue that minimizing SelfCon loss maximizes the lower bound of MI, which results in the improved classification performance presented in Section 4. To empirically confirm this claim, we design an interpolation between SupCon and SelfCon-M loss as follows:

\[
\mathcal{L}_{\omega \leftarrow \mathcal{L}} = \frac{1}{1 + \alpha} \mathcal{L}_\text{sup} + \frac{\alpha}{1 + \alpha} \bigg|_{\omega = \mathcal{L}}
\]

If \( \alpha = 0 \), \( \mathcal{L}_{\omega \leftarrow \mathcal{L}} \) is equivalent to the SupCon loss, and if \( \alpha = 1 \), then \( \mathcal{L}_{\omega \leftarrow \mathcal{L}} \) is almost the same as SelfCon-M loss. We cannot make the exact interpolation because SelfCon-M has contrastive pairs from the sub-network, whereas SupCon does not.

We estimated MI using various estimators: InfoNCE (Oord et al., 2018), MINE (Belghazi et al., 2018), and NWJ

Figure 2. Test accuracy and the estimated mutual information of different methods. SelfCon-M*\( \alpha \) denotes SelfCon-M* loss with hyperparameter \( \alpha \). We measured the mutual information estimators between the intermediate and the last features. We used ResNet-18 on the CIFAR-100 dataset for the measurements. When \( \alpha \geq 0.2 \), the test accuracy was similar to that of SelfCon-M.

(Nguyen et al., 2010). We measured \( \mathcal{I}(F(x);T(x)) \) because it is difficult to estimate the conditional MI. We observed a clear increasing trend for both MI and the test accuracy as the contribution of SelfCon becomes larger (i.e., increasing \( \alpha \)). After SelfCon loss increases the correlation between \( F(x) \) and \( T(x) \), the rich information in earlier features enables the encoder to output a better representation because the intermediate feature is also the input for the subsequent layers. Refer to Appendix E for a detailed SelfCon-M* loss formulation and the exact numbers.

5.3. Single-view vs. Multi-view

Single-view is efficient in terms of memory usage and computational cost. In Table 5, we compare SupCon and SelfCon-S to observe the efficiency of a single-viewed batch, varying the scales of the benchmarks. For both SupCon and SelfCon-S, the same batch size implies the same number of anchor features; however, SupCon consumes nearly twice the memory cost compared with SelfCon-S due to the data augmentation of the multi-viewed batch. Although SelfCon-S requires a larger number of parameters owing to the extra sub-network, its memory and time consumption in practice are much more efficient. Our work can guide work for efficient contrastive learning (Koohpayegani et al., 2020; Fang et al., 2021).

Single-view reduces generalization error. In Figure 3, SupCon shows higher train accuracy, but lower test accuracy than SupCon-S, and the same trend is observed with SelfCon-M and SelfCon-S (blue vs. red). Compared with single-view, multi-view from the augmented image makes the encoder amplify the memorization of data and results in overfitting to each instance. In addition, Figure 4 shows that
SelfCon-loss regularizes the sub-network to output similar features to the backbone network. It prevents the encoder from overfitting the data, and it is effective in multi-viewed as well as single-viewed batches. In Figure 3, we confirm the regularization effect (i.e., lower train accuracy, but higher test accuracy) by comparing each bar of the same color. The strong regularization of the subnetwork helped SelfCon (-M, -S) outperform the SupCon counterparts. This trend can also be observed in Figure 4 and Table 6.

**Table 6. CIFAR-100 results on ResNet-18 with various batch sizes.**

| Method     | 64  | 128 | 256 | 512 | 1024 |
|------------|-----|-----|-----|-----|------|
| CE         | 74.9| 74.9| 74.1| 73.3| 72.9 |
| SupCon     | 74.8| 73.8| 72.9| 72.5| 73.0 |
| SupCon-S   | 73.6| 75.3| 75.0| 74.0| 73.9 |
| SelfCon-M  | 75.8| 76.5| 75.9| 75.0| 74.9 |
| SelfCon-S  | 74.0| 76.6| 77.0| 75.8| 75.4 |

**Figure 3. Train accuracy and test accuracy on ResNet-18 for different views and loss functions.** The train and test accuracy are measured with a linear classifier during the linear evaluation. The axis on the left and right denotes the train accuracy and test accuracy, respectively.

SelfCon-S gradually enhances generalization ability, while SelfCon-M and SupCon achieve a little gain in test accuracy despite the fast convergence.

**Multi-view is advantageous for small batch size.** In supervised learning, a large batch size has been known to reduce generalization ability, which degrades performance (You et al., 2017; Luo et al., 2018; Wu et al., 2020). We examined whether the performance in a supervised contrastive framework is also dependent on the batch size. Table 6 summarizes the results for CIFAR-100 benchmark. SelfCon-S showed the best performance in every case except for the batch size of 64. However, the multi-viewed method outperformed the single-viewed counterpart in 64-batch experiments, where underfitting may occur because of large randomness from the small batch size or the small number of positive pairs. In the ImageNet experiments (see Table 3), SelfCon-M consistently outperformed every method, implying that it is more important to mitigate underfitting. As a result, multi-viewed methods show better performance in the underfitting scenario (e.g., small batch size or large-scale benchmark).

**5.4. What Does the Sub-network Achieve?**

**Regularization effect**  SelfCon loss regularizes the sub-network to output similar features to the backbone network. It prevents the encoder from overfitting the data, and it is effective in multi-viewed as well as single-viewed batches. In Figure 3, we confirm the regularization effect (i.e., lower train accuracy, but higher test accuracy) by comparing each bar of the same color. The strong regularization of the subnetwork helped SelfCon (-M, -S) outperform the SupCon counterparts. This trend can also be observed in Figure 4 and Table 6.

**Ensemble with sub-network**  The sub-network in SelfCon learning can also be used on downstream tasks such as image classification. We followed the linear evaluation protocol in the representation learning experiments by fine-tuning a classifier with the frozen backbone network. The network pretrained on SelfCon learning has an exit path, which is similarly allowed to be frozen and used as linear evaluation. Training an extra classifier after the frozen subnetwork does not demand a high cost in the fine-tuning
**Table 7.** Classification accuracy with the classifiers after backbone, sub-network, and the ensemble of them. The ResNet-18 encoder is pretrained by the SelfCon-S loss function. CF and IN indicate CIFAR and ImageNet, respectively. Refer to Appendix G for the results on ResNet-50.

| Method  | CF-10 | CF-100 | Tiny-IN | IN-100 | IN  |
|---------|-------|--------|---------|--------|-----|
| SupCon  | 94.7  | 73.0   | 56.9    | 85.5   | 71.2|
| Backbone| **95.3** | 75.4 | 59.8 | 86.3 | 71.5 |
| Sub-network | 92.6 | 69.1 | 53.5 | 85.0 | 71.2 |
| Ensemble | 95.2 | **77.4** | **62.2** | **87.2** | **72.6** |

**Figure 6.** Gradient norm of each ResNet-18 block and convolutional layer. We computed gradients from the SupCon loss (**Left**) and SelfCon-M loss (**Right**), both from the same initialized model. All convolution layers in the block are named by order.

**Zhang et al., 2019a** have pointed out that the success of the multi-exit framework owes to solving the vanishing gradient problem. We show that the same argument applies to our SelfCon learning. Note that the sub-network is positioned after the 2nd block of the ResNet-18 backbone network. In Figure 6, a large gradient flows up to the earlier layer in the SelfCon-M, whereas a large amount of the SupCon loss gradient vanishes. In particular, there is a significant difference in the gradient norm in the 2nd block of the encoder of SupCon and SelfCon-M.

### 6. Conclusion

We have proposed an efficient supervised contrastive framework called Self-Contrastive learning. By contrasting the features from multiple levels of a network, SelfCon learning can be free from the issues, e.g., high computational cost and generalization error, that a multi-viewed batch involves. We theoretically prove that SelfCon loss regularizes the intermediate features to learn the label information in the last feature, improving the classification performance. In addition, we analyze why SelfCon learning is better than SupCon by exploring the effect of single-view and sub-network, such as the regularization effect, computational efficiency, or ensemble prediction. We verify by extensive experiments, including ImageNet-100 and ImageNet, that SelfCon-S loss outperforms CE and SupCon loss.
References

Ahn, S., Hu, S. X., Damianou, A., Lawrence, N. D., and Dai, Z. Variational information distillation for knowledge transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9163–9171, 2019.

Bachman, P., Hjelm, R. D., and Buchwalter, W. Learning representations by maximizing mutual information across views. arXiv preprint arXiv:1906.00910, 2019.

Barber, D. and Agakov, F. The im algorithm: a variational approach to information maximization. Advances in neural information processing systems, 16(320):201, 2004.

Belghazi, M. I., Baratin, A., Rajeswar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, R. D. Mine: mutual information neural estimation. arXiv preprint arXiv:1801.04062, 2018.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. arXiv preprint arXiv:2006.09882, 2020.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. arXiv preprint arXiv:2104.14294, 2021.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In International conference on machine learning, pp. 1597–1607. PMLR, 2020.

Chen, X. and He, K. Exploring simple siamese representation learning. arXiv preprint arXiv:2011.10566, 2020.

Cover, T. M. Elements of information theory. John Wiley & Sons, 1999.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 113–123, 2019.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 702–703, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pp. 248–255. Ieee, 2009.

Fang, Z., Wang, J., Wang, L., Zhang, L., Yang, Y., and Liu, Z. Seed: Self-supervised distillation for visual representation. arXiv preprint arXiv:2101.04731, 2021.

Goldfeld, Z., van den Berg, E., Greenewald, K. H., Melnyk, I., Nguyen, N., Kingsbury, B., and Polyanskiy, Y. Estimating information flow in deep neural networks. In ICML, 2019.

Goyal, P., Dollár, P., Girshick, R., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:1706.02677, 2017.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., et al. Bootstrap your own latent: A new approach to self-supervised learning. arXiv preprint arXiv:2006.07733, 2020.

Gunel, B., Du, J., Conneau, A., and Stoyanov, V. Supervised contrastive learning for pre-trained language model fine-tuning. arXiv preprint arXiv:2011.01403, 2020.

Gutmann, M. and Hyvärinen, A. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 297–304. JMLR Workshop and Conference Proceedings, 2010.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9729–9738, 2020.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., and Bengio, Y. Learning deep representations by mutual information estimation and maximization. arXiv preprint arXiv:1808.06670, 2018.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, pp. 448–456. PMLR, 2015.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. arXiv preprint arXiv:2004.11362, 2020.

Kim, S., Lee, G., Bae, S., and Yun, S.-Y. Mixco: Mix-up contrastive learning for visual representation. arXiv preprint arXiv:2010.06300, 2020.
Koohpayegani, S. A., Tejankar, A., and Pirsiavash, H. Compress: Self-supervised learning by compressing representations. arXiv preprint arXiv:2010.14713, 2020.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lee, Y. and Yang, X. Tiny imagenet visual recognition challenge. CS 231N, 7:7, 2015.

Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. Deeply-supervised nets. In Artificial intelligence and statistics, pp. 562–570. PMLR, 2015.

Lee, K., Zhu, Y., Sohn, K., Li, C.-L., Shin, J., and Lee, H. I-mix: A domain-agnostic strategy for contrastive representation learning. arXiv preprint arXiv:2010.08887, 2020.

Linsker, R. An application of the principle of maximum information preservation to linear systems. In Advances in neural information processing systems, pp. 186–194, 1989.

Loshchilov, I. and Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983, 2016.

Luo, P., Wang, X., Shao, W., and Peng, Z. Towards understanding regularization in batch normalization. arXiv preprint arXiv:1809.00846, 2018.

Nguyen, X., Wainwright, M. J., and Jordan, M. I. Estimating divergence functionals and the likelihood ratio by convex risk minimization. IEEE Transactions on Information Theory, 56(11):5847–5861, 2010.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.

Paninski, L. Estimation of entropy and mutual information. Neural computation, 15(6):1191–1253, 2003.

Poole, B., Ozair, S., Van Den Oord, A., Alemi, A., and Tucker, G. On variational bounds of mutual information. In International Conference on Machine Learning, pp. 5171–5180. PMLR, 2019.

Rasmus, A., Valpola, H., Honkala, M., Berglund, M., and Raiko, T. Semi-supervised learning with ladder networks. arXiv preprint arXiv:1507.02672, 2015.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. On the information bottleneck theory of deep learning. Journal of Statistical Mechanics: Theory and Experiment, 2019(12):124020, 2019a.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., and Cox, D. D. On the information bottleneck theory of deep learning. Journal of Statistical Mechanics: Theory and Experiment, 2019(12):124020, 2019b.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision, pp. 618–626, 2017.

Shwartz-Ziv, R. and Tishby, N. Opening the black box of deep neural networks via information. arXiv preprint arXiv:1703.00810, 2017.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

Song, J. and Ermon, S. Understanding the limitations of variational mutual information estimators. arXiv preprint arXiv:1910.06222, 2019.

Song, J. and Ermon, S. Multi-label contrastive predictive coding. arXiv preprint arXiv:2007.09852, 2020.

Sordoni, A., Dziri, N., Schulz, H., Gordon, G., Bachman, P., and Des Combes, R. T. Decomposed mutual information estimation for contrastive representation learning. In International Conference on Machine Learning, pp. 9859–9869. PMLR, 2021.

Teerapittayanon, S., McDanel, B., and Kung, H.-T. Branchynet: Fast inference via early exiting from deep neural networks. In 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 2464–2469. IEEE, 2016.

Tian, Y., Krishnan, D., and Isola, P. Contrastive multiview coding. arXiv preprint arXiv:1906.05849, 2019a.

Tian, Y., Krishnan, D., and Isola, P. Contrastive representation distillation. arXiv preprint arXiv:1910.10699, 2019b.

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning. arXiv preprint arXiv:2005.10243, 2020.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In 2015 IEEE Information Theory Workshop (ITW), pp. 1–5. IEEE, 2015.

Verma, V., Luong, T., Kawaguchi, K., Pham, H., and Le, Q. Towards domain-agnostic contrastive learning. In International Conference on Machine Learning, pp. 10530–10541. PMLR, 2021.
Self-Contrastive Learning: An Efficient Supervised Contrastive Framework with Single-view and Sub-network

Wang, W., Zhou, T., Yu, F., Dai, J., Konukoglu, E., and Van Gool, L. Exploring cross-image pixel contrast for semantic segmentation. *arXiv preprint arXiv:2101.11939*, 2021.

Wu, J., Hu, W., Xiong, H., Huan, J., Braverman, V., and Zhu, Z. On the noisy gradient descent that generalizes as sgd. In *International Conference on Machine Learning*, pp. 10367–10376. PMLR, 2020.

Yeung, R. W. A new outlook on shannon’s information measures. *IEEE transactions on information theory*, 37(3):466–474, 1991.

You, Y., Gitman, I., and Ginsburg, B. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhang, L., Song, J., Gao, A., Chen, J., Bao, C., and Ma, K. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3713–3722, 2019a.

Zhang, L., Tan, Z., Song, J., Chen, J., Bao, C., and Ma, K. Scan: A scalable neural networks framework towards compact and efficient models. *arXiv preprint arXiv:1906.03951*, 2019b.
Appendix

A. Proofs

A.1. Proof of Proposition 4.1

Proof. We extend the exact bound of InfoNCE (Poole et al., 2019; Sordoni et al., 2021). Here, we consider the supervised setting where there are $C$ training classes. Without loss of generality, choose a class $c$ out of $C$ classes, and let $x$ and $z$ be different samples that share the same class label $c$. The derivation for the multi-view ($z$ being an augmented sample of $x$) is similar. For conciseness of the proof, we consider that no other image in a batch shares the same class. We prove that minimizing the SupCon loss (Khosla et al., 2020) maximizes the lower bound of conditional MI between two samples $x$ and $z$ given the label $c$:

$$\mathcal{I}(x; z|c) \geq \log(2K - 1) - \mathcal{L}_{sup}(x, z; F, K)$$

(8)

for some function $F$ and hyperparameter $K$.

We start from Barber and Agakov’s variational lower bound on MI (Barber & Agakov, 2004).

$$\mathcal{I}(x; z|c) = \mathbb{E}_{p(x, z|c)} \log \frac{p(z|x, c)}{p(z|c)} \geq \mathbb{E}_{p(x, z|c)} \log \frac{q(z|x, c)}{p(z|c)}$$

(9)

where $q$ is a variational distribution. Since $q$ is arbitrary, we can set the sampling strategy as follows. First, sample $z_1$ from the proposal distribution $\pi(z|c)$ where $c$ is a class label of $x$. Then, sample $(K - 1)$ negative samples $\{z_2, \cdots, z_K\}$ from the distribution $\sum_{c' \neq c} \pi(z|c')$, so that these negative samples do not share the class label with $x$. We augment each negative sample by random augmentation and concatenate with the original samples, i.e., $\{z_2, \cdots, z_K, z_{K+1}, \cdots, z_{2K-1}\}$, where $z_{K+i-1}$ is the augmented sample from $z_i$ for $2 \leq i \leq K$. We define the unnormalized density of $z_1$ given a specific set $\{z_2, \cdots, z_{2K-1}\}$ and $x$ of label $c$ is

$$q(z_1|x, z_{2:(2K-1)}, c) = \pi(z_1|c) \cdot \frac{(2K - 1) \cdot e^{\psi_F(z_1, x)}}{\sum_{k=2}^{2K-1} e^{\psi_F(z_k, x)}}$$

(10)

where $\psi$ is often called a discriminator function (Hjelm et al., 2018), defined as $\psi_F(u, v) = F(u) \cdot F(v)$ for some vectors $u, v$. By setting the proposal distribution as $\pi(z|c) = p(z|c)$, we obtain the MI bound:

$$\mathcal{I}(x; z|c) \geq \mathbb{E}_{p(x, z_1|c)} \log \frac{q(z_1|x, c)}{p(z_1|c)}$$

(11)

$$= \mathbb{E}_{p(x, z_1|c)} \log \frac{\mathbb{E}_{p(z_2:(2K-1)|c)} q(z_1|x, z_{2:(2K-1)}, c)}{p(z_1|c)}$$

(12)

$$\geq \mathbb{E}_{p(x, z_1|c)} \left[ \log \frac{p(z_1|c) \cdot (2K - 1) \cdot e^{\psi_F(z_1, x)}}{\sum_{k=2}^{2K-1} e^{\psi_F(z_k, x)}} \right]$$

(13)

$$= \mathbb{E}_{p(x, z_1|c)} \log \frac{1}{2K - 1} \sum_{k=1}^{2K-1} e^{\psi_F(z_k, x)}$$

(14)

$$= \log(2K - 1) - \mathcal{L}_{sup}(x, z; F, K).$$

(15)

where the second inequality is derived from Jensen’s inequality. Because Eq. 14 is an expectation with respect to the sampled $x$ and $z_1$, the case where the anchor is swapped to $z_1$ is also being considered.

A neural network $F$ (backbone in our framework) with $L$ layers are formulated as $F = f_L \circ f_{L-1} \circ \cdots \circ f_1$. Then, $\psi_F(u, v) = F(u) \cdot F(v) = f_{1:L}(u) \cdot f_{1:L}(v)$. We define another discriminator function as $\psi_F(u, v) = f_{(t+1):L}(u) \cdot f_{(t+1):L}(v)$. 

We introduce a discriminator function that measures the similarity between the outputs from the backbone and the sub-network: 

\[ I(f_{1, \ell}(u), f_{1, \ell}(v)) = \psi_{F}(u, v). \]  

(16)

Note that \( f_{1, \ell}(u) \) is the \( \ell \)-th intermediate feature of input \( u \). Following the same procedure as in Eq. 11-15,

\[ I(f_{1, \ell}(x); f_{1, \ell}(z)) \geq \mathbb{E}_p(x, z | c) \log \frac{e^{\psi_{F}(f_{1, \ell}(x), f_{1, \ell}(z))}}{\frac{1}{2K-1} \sum_{k=1}^{2K-1} e^{\psi_{F}(f_{1, \ell}(x), f_{1, \ell}(z_k))}} \]

(17)

\[ = \mathbb{E}_p(x, z | c) \log \frac{1}{2K-1} \sum_{k=1}^{2K-1} e^{\psi_{F}(x, z_k)} \]

(18)

\[ = \log(2K - 1) - \mathcal{L}_{\text{sup}}(x, z; F, K). \]

(19)

From above, as the intermediate feature is arbitrary to the position, we can obtain a similar inequality:

\[ I(f_{(\ell+1); L}(f_{1, \ell}(x)); f_{(\ell+1); L}(f_{1, \ell}(z))) \geq \log(2K - 1) - \mathcal{L}_{\text{sup}}(x, z; F, K). \]

(22)

A.2. Proof of Proposition 4.2

Proof. In Section 4.1, we proved that SupCon loss maximizes the lower bound of conditional MI between the output features of a positive pair. We can think of another scenario where the network \( F \) now has a sub-network \( G \). Assume that the sub-network has \( M > \ell \) layers: \( G = g_M \circ g_{M-1} \circ \cdots \circ g_1 \). As we discussed in the paper, the exit path is placed after the \( \ell \)-th layer, so regarding our definition of the sub-network, \( G \) shares the same parameters with \( F \) up to \( \ell \)-th layer, i.e., \( g_1 = f_1, g_2 = f_2, \cdots, g_\ell = f_\ell \). Define \( \psi_G(u, v) = G(u) \cdot G(v) \)

We introduce a discriminator function that measures the similarity between the outputs from the backbone and the sub-network, \( \psi_{FG}(u, v) = F(u) \cdot G(v) \). Similarly, \( \psi_{GF}(u, v) = G(u) \cdot F(v) \). Considering that the SelfCon-S loss has the anchored function of \( F \) and \( G \), we obtain an upper bound of two symmetric mutual information. Here, \( z_1 = x \) because SelfCon-S loss is defined on the single-viewed batch and we assume that other images in a batch (i.e., \( z_2, \cdots, z_K \)) are sampled from the different class label with \( x \).

\[ I(F(x); G(x)|c) + I(G(x); F(x)|c) \]

\[ \geq \mathbb{E} \log \frac{e^{\psi_{FG}(x, x)}}{\frac{1}{2K-1} (e^{\psi_{FG}(x, x)} + \sum_{k=2}^{K} e^{\psi_{FG}(x, z_k)} + \sum_{k=2}^{K} e^{\psi_{FG}(x, z_k)})} \]

(23)

\[ + \mathbb{E} \log \frac{e^{\psi_{GF}(x, x)}}{\frac{1}{2K-1} (e^{\psi_{GF}(x, x)} + \sum_{k=2}^{K} e^{\psi_{GF}(x, z_k)} + \sum_{k=2}^{K} e^{\psi_{GF}(x, z_k)})} \]

(24)

\[ = 2 \log(2K - 1) - 2 \mathcal{L}_{\text{self-s}}(x; \{ F, G \}, K) \]

(25)

Due to the symmetry of mutual information,

\[ I(F(x); G(x)|c) \geq \log(2K - 1) - \mathcal{L}_{\text{self-s}}(x; \{ F, G \}, K) \]

(26)
In addition, we can similarly bound the SelfCon-M loss. As the derivation of SupCon loss bound, only consider the anchor \( x \) and its positive pair \( z_1 \). When the anchored feature is \( F(x) \), the contrastive features are: \( G(x), G(z), \) and \( F(z) \). By symmetry, when the anchored feature is \( G(x) \), the contrastive features are: \( F(x), F(z), \) and \( G(z) \). As the derivation of the SupCon loss bound, we assume the augmented negative samples, i.e., \( \{z_2, \ldots, z_{K+1}, \ldots, z_{2K-1}\} \).

\[
\begin{align*}
\mathcal{I}(F(x); G(x)|c) + \mathcal{I}(F(x); F(z)|c) + \mathcal{I}(G(x); F(x)|c) + \mathcal{I}(G(x); G(z)|c)
&\geq \frac{1}{3} \mathbb{E} \log \frac{1}{4K-1} \left( e^{\psi_{FG}(x,x)} + \sum_{k=1}^{2K-1} e^{\psi_{FG}(x,z_k)} + \sum_{k=1}^{2K-1} e^{\psi_{FG}(z_k,z_k)} \right) \\
&+ \frac{1}{3} \mathbb{E} \log \frac{1}{4K-1} \left( e^{\psi_{GF}(x,x)} + \sum_{k=1}^{2K-1} e^{\psi_{GF}(x,z_k)} + \sum_{k=1}^{2K-1} e^{\psi_{GF}(z_k,z_k)} \right) \\
&= \frac{2}{3} \log(4K-1) - 2\mathcal{L}_{self,f.m.}(x,z; \{F,G\}, K)
\end{align*}
\]

There could be a doubt about the loose bound between SelfCon loss and MI. However, when we prove the MI bound, we assumed a probabilistic model (refer to Eq. 10). When the anchor feature is similar to the negative pairs (i.e., different class representations), this model becomes a variational distribution with random mapping, and SelfCon loss cannot be optimized at all. Therefore, optimizing SelfCon loss means that the representations of different classes get farther. Then, a better estimation of variational distribution leads to a small gap between SelfCon loss and MI. After all, SelfCon loss has improved performance because it tightens the bound of the label-conditional MI while distinguishing different class representations.

A.3. Proof of Proposition 4.3

\( F(x) \) and \( G(x) \) are the output features from the backbone network and the sub-network, respectively. Recall that \( T \) denotes the sharing layers between \( F \) and \( G \). \( T(x) \) is the intermediate feature of the backbone, which is also an input to the auxiliary network path.

Before proving Proposition 4.3, we would like to note that the usefulness of mutual information should be carefully discussed on the stochastic mapping of a neural network. If a mapping \( T(x) \mapsto F(x) \) is a deterministic mapping, then the MI between \( T(x) \) and \( F(x) \) is degenerate because \( \mathcal{I}(T(x); F(x)) \) is either infinite for continuous \( T(x) \) (conditional differential entropy is \( -\infty \)) or a constant for discrete \( T(x) \) which is independent on the network’s parameters (equal to \( \mathcal{H}(T(x)) \)). However, for studying the usefulness of mutual information in a deep neural network, the map \( T(x) \mapsto F(x) \) is considered as a stochastic parameterized channel. In many recent works about information theory with DNN, they view the training via SGD as a stochastic process, and the stochasticity in the training procedure lets us define the MI with stochastically trained representations (Schwartz-Ziv & Tishby, 2017; Goldfeld et al., 2019; Saxe et al., 2019b; Goldfeld et al., 2019). Our theoretical claim focuses on the SelfCon loss as a training loss optimized by the SGD algorithm. Therefore, analyzing the MI between the hidden representations while training with the SelfCon loss is based on the information theory to understand DNN (Tishby & Zaslavsky, 2015).

Also, information theory in deep learning, especially in contrastive learning, is based on the InfoMax principle (Linsker, 1989) which is about learning a neural network that maps a set of input to a set of output to maximize the average mutual information between the input and output of a neural network, subject to stochastic processes. This InfoMax principle is nowadays widely used for analyzing and optimizing DNNs. Most works for contrastive learning are based on maximizing mutual information grounds on the InfoMax principle, and they are grounded on the stochastic mapping of an encoder. Moreover, Poole et al. (2019) rigorously discussed the mutual information with respect to a stochastic encoder. This is common practice in a representation learning context where \( x \) is data, and \( z \) is a learned stochastic representation.

\[
\mathcal{I}(F(x); G(x)|c) \leq \mathcal{I}(F(x); T(x)|c)
\]

**Proof.** As \( F(x) \) and \( G(x) \) are conditionally independent given the intermediate representation \( T(x) \), they formulate a Markov chain as follows: \( G \leftrightarrow T \leftrightarrow F \) (Cover, 1999). Under this relation, the following is satisfied:
\[ \mathcal{I}(F(x); G(x)|c) = \mathcal{H}(F(x)|c) - \mathcal{H}(F(x)|G(x), c) \]  
\[ \leq \mathcal{H}(F(x)|c) - \mathcal{H}(F(x)|T(x), G(x), c) \]  
\[ = \mathcal{H}(F(x)|c) - \int_{t,f,g} p(t,f,g|c) \log p(f|t,g,c) dt df dg \]  
\[ = \mathcal{H}(F(x)|c) - \int_{t,f} p(t,f|c) \log p(f|t,c) dt df \]  
\[ = \mathcal{I}(F(x); T(x)|c) \]  
\[ \mathcal{I}(F(x); G(x)|c) \]

Eq. 32 is from the property of conditional entropy, and Eq. 34 is due to the conditional independence and marginalization of \( g \).

From Eq. 30 and Eq. 26, we further obtain Eq. 6 as follows:

\[ \log(2K - 1) - \mathcal{L}_{self,s}(x; \{F, G\}, K) \]  
\[ \leq \mathcal{I}(F(x); G(x)|c) \]  
\[ \leq \mathcal{I}(F(x); T(x)|c) \]  
\[ = \mathcal{I}(F(x); T(x), c) - \mathcal{I}(F(x); c) \]  
\[ = \mathcal{I}(F(x); T(x)) + \mathcal{I}(F(x); c|T(x)) - \mathcal{I}(F(x); c) \]

Strictly speaking, SelfCon-S loss does not guarantee the lower bound of either (\( \square \)) or (\( \blacksquare \)) in Eq. 41. However, SelfCon-S loss guarantees the label-conditional MI between the intermediate and the last feature, which is (\( \square + \blacksquare \)).

**B. Implementation Details**

**B.1. Network Architectures**

We modified the architecture of networks according to the benchmarks. For the smaller scale of benchmarks (e.g., CIFAR-10, CIFAR-100, and Tiny-ImageNet) and the residual networks (e.g., ResNet-18, ResNet-50, and WRN-16-8), we changed the kernel size and stride of a convolution head to 3 and 1, respectively. We also excluded Max-Pooling on the top of the ResNet architecture for the CIFAR datasets. Moreover, for VGG-16 with BN, the dimension of the fully-connected layer was changed from 4096 to 512 for CIFAR and Tiny-ImageNet. MLP projection head for contrastive learning consisted of two convolution layers with 128 dimensions and one ReLU activation. For the architectures of sub-networks, refer to Appendix B.4.

**B.2. Representation Learning**

We refer to the technical improvements used in SupCon, i.e., a cosine learning rate scheduler (Loshchilov & Hutter, 2016), an MLP projection head (Chen et al., 2020), and the augmentation strategies (Cubuk et al., 2019): {ResizedCrop, HorizontalFlip, ColorJitter, GrayScale}. ColorJitter and GrayScale are only used in the pretraining stage. For small-scale benchmarks, we used 8 GPUs and set the batch size to 1024 for the pretraining and 512 for the linear evaluation. We trained the encoder and the linear classifier for 1000 epochs and 100 epochs, respectively. Meanwhile, we used the batch size of 512 when pretraining on the ImageNet-100 dataset. Besides, we trained the encoder and the linear classifier for 400 epochs and 40 epochs, respectively, in ImageNet-100 and ImageNet benchmark.

Every experiment used SGD with 0.9 momentum and weight decay of 1e-4 without Nesterov momentum. All contrastive loss functions used temperature \( \tau \) of 0.1. For a fair comparison to (Khosla et al., 2020), we set the same learning rate of the encoder network as 0.5 for the small-scale benchmarks. Refer to Appendix B.4 for the optimal learning rate of the
large-scale dataset. We linearly scaled the learning rate according to the batch size (Goyal et al., 2017). On the linear evaluation step, we used 5.0 as a learning rate of the linear classifier for the residual architecture, but it was robust to any value and converged in nearly 20 epochs. Meanwhile, for VGG architecture, only a small learning rate of 0.1 converged.

B.3. 1-Stage Training

In the 1-stage training protocol, we trained the encoder network jointly with a linear classifier on the single-viewed batch. Most of the experimental settings were the same as those of representation learning, but we trained the encoder for 500 epochs with a cosine learning rate scheduler on all benchmarks except ImageNet. We used the batch size of 1024 and the learning rate of 0.8 for small-scale datasets, and 512 batch size and 0.4 learning rate for ImageNet-100. For the cross-entropy result of the ImageNet dataset, we trained for 90 epochs with the multi-step learning rate scheduler after 30 and 60 epochs with the decay ratio of 0.1.

In the multi-exit framework, we used a linear combination of loss functions for the backbone and sub-network. We used only cross-entropy loss for the backbone network and weighted linear combinations of the loss functions (e.g., KL divergence and SelfCon-S) for the sub-network. For example, Self-Distillation (Zhang et al., 2019a) used the interpolation coefficient \( \alpha \) of 0.5. For the 1-stage version of SelfCon loss, we follow the coefficient form in (Tian et al., 2019b): \( L = L_{CE} + \beta L_{self-s} \). We set the coefficient \( \beta = 0.8 \) for all experiments. Note that we used the outputs from the projection head instead of the logits. We did not use the interpolated form, unlike SD, because we distill the features from the projection head. We used temperature \( \tau = 3.0 \) for SD and \( \tau = 0.1 \) for SelfCon loss.

B.4. Hyperparameters

Sensitivity study for learning rate In Table 6, we experimented with the supervised contrastive algorithms with various batch sizes and confirmed that the classification accuracy decreases in the large batch size. We supposed that this trend is induced by the regularization effect from the batch size. However, there could be a concern for using a sub-optimal learning rate on the large batch size.

We further studied the sensitivity for the learning rate in a batch size of 1024 on CIFAR-100 and summarized the results in Table 8. We concluded that the performance comparison in Table 6 is consistent with hyperparameter tuning. The experimental results supported that a larger learning rate than 0.5 may be a better choice but the trend between all methods maintained in parallel with the learning rate of 0.5. Therefore, we stick to the initial learning rate of 0.5 that Khosla et al. (2020) had used.

Moreover, we tuned the learning rate for the reliability of our large-scale experiments in ImageNet-100 and ImageNet datasets. Table 9 summarized the sensitivity results in ImageNet-100 on ResNet-18 architecture. In this experiment, we fixed the batch size to 512 and pretraining epochs to 400. Note that although we used the same sub-network in this experiment, Table 2 reports the results with the small sub-network after our sub-network experiments in the next section. We confirmed that the learning rate of 0.5 is the best except for SupCon, which showed the best at 1.0. Every other experiment in ImageNet-100 used this best setting.

Table 10 summarized the results in ImageNet on ResNet-18 architecture. We fixed the batch size to 3072 and pretraining epochs to 400. We compared SelfCon-S with SupCon, two major methods in our paper, and confirmed that the learning rate of 0.375 is the best. We used 0.375 for other experiments, and we used a linear scaling rule for other batch sizes (e.g., 0.25 for 2048 batch size and 0.125 for 1024 batch size). The full
Table 10. **ImageNet results on ResNet-18 with various learning rates. Bold** type is for the best accuracy within each method. We used a *same* sub-network structure for SelfCon-S.

| Method   | Learning Rate |       |       |       |
|----------|---------------|-------|-------|-------|
|          | 0.1875        | 0.375 | 0.75  | 1.5   |
| SupCon   | 71.0          | 71.2  | 71.2  | 70.7  |
| SelfCon-S| 71.3          | 71.3  | 71.1  | 69.1  |

Table 11. **The results of SelfCon-S loss according to the structure and position of sub-network.** The classification accuracy is for ResNet-18 *(Left)* and ResNet-50 *(Right)* on the CIFAR-100 benchmark.

| Structure | Position | Accuracy |
|-----------|----------|----------|
|           | 1\textsuperscript{st} Block | 2\textsuperscript{nd} Block | 3\textsuperscript{rd} Block |
| FC        | ✓        | ✓        | ✓        | 75.4±0.1 |
| Small     | ✓        | ✓        | ✓        | 74.7±0.2 |
| Same      | ✓        | ✓        | ✓        | 74.5±0.0 |
| FC        | ✓        | ✓        | ✓        | 73.2±0.2 |
| FC        | ✓        | ✓        | ✓        | 75.4±0.1 |

| Structure | Position | Accuracy |
|-----------|----------|----------|
|           | 1\textsuperscript{st} Block | 2\textsuperscript{nd} Block | 3\textsuperscript{rd} Block |
| FC        | ✓        | ✓        | ✓        | 78.5±0.3 |
| Small     | ✓        | ✓        | ✓        | 78.1±0.2 |
| Same      | ✓        | ✓        | ✓        | 77.4±0.2 |
| FC        | ✓        | ✓        | ✓        | 77.0±0.2 |
| FC        | ✓        | ✓        | ✓        | 78.5±0.3 |
| FC        | ✓        | ✓        | ✓        | 77.4±0.1 |
| FC        | ✓        | ✓        | ✓        | 78.7±0.5 |

Table 12. **The results of SelfCon-S loss according to the structure and position of sub-network.** The classification accuracy is for WRN-16-8 *(Left)* and VGG-16 with BN *(Right)* on the CIFAR-100 benchmark.

| Structure | Position | Accuracy |
|-----------|----------|----------|
|           | 1\textsuperscript{st} Block | 2\textsuperscript{nd} Block |
| FC        | ✓        | ✓        | 74.4±1.2 |
| Small     | ✓        | ✓        | 76.2±0.0 |
| Same      | ✓        | ✓        | 76.6±0.1 |
| Same      | ✓        | ✓        | 76.5±0.2 |
| Same      | ✓        | ✓        | 76.5±0.0 |

| Structure | Position | Accuracy |
|-----------|----------|----------|
|           | 1\textsuperscript{st} Block | 2\textsuperscript{nd} Block | 3\textsuperscript{rd} Block | 4\textsuperscript{th} Block |
| FC        | ✓        | ✓        | ✓        | ✓        | 71.4±0.0 |
| Small     | ✓        | ✓        | ✓        | ✓        | 71.5±0.4 |
| Same      | ✓        | ✓        | ✓        | ✓        | 71.5±0.3 |
| FC        | ✓        | ✓        | ✓        | ✓        | 70.9±0.1 |
| FC        | ✓        | ✓        | ✓        | ✓        | 71.4±0.0 |
| FC        | ✓        | ✓        | ✓        | ✓        | 72.0±0.0 |
| FC        | ✓        | ✓        | ✓        | ✓        | 71.5±0.1 |
| FC        | ✓        | ✓        | ✓        | ✓        | 72.5±0.1 |

The results for 3072 batch size are summarized in Appendix C.2.

**Sub-network** The structure, position, and number of sub-networks are important to the performance of SelfCon learning. First, in order to find a suitable structure of the sub-network, the following three structures were attached after the 2\textsuperscript{nd} block of an encoder: (1) a simple fc, fully-connected, layer, (2) small structure which reduced by half the number of layers in the non-sharing blocks, (3) same structure which is same as the backbone’s non-sharing block structure. After we found the optimal structure, we fixed the structure of the sub-network and found which position was the best. For ResNet architectures, there are three positions to attach; after the 1\textsuperscript{st}, 2\textsuperscript{nd}, and 3\textsuperscript{rd} block. For VGG-16 with BN, there are four positions, and for WRN-16-8, there are two positions possible. Note that blocks are divided based on the Max-Pooling layer in VGG-16 with BN.

Table 11 presents the ablation study results for ResNet-18 and ResNet-50, and Table 12 presents the results for WRN-16-8 and VGG-16 with BN. We observed a trend: in shallow networks (e.g., WRN-16-8) the same structure was better, while fc was better in deeper networks (e.g., VGG-16 with BN and ResNet). Besides, the performance was consistently good when the exit path is attached after the midpoint of the encoder (e.g., 2\textsuperscript{nd} block in ResNet or 3\textsuperscript{rd} block in VGG architecture). We highlighted the selected structure and position in Table 11 and Table 12.

Obviously, there are many combinations of placing sub-networks, and Table 11 and Table 12 presented an interesting result that some performance was the best when sub-networks were attached to all blocks. It seems that increasing the number of
positive and negative pairs by various views from multiple sub-networks improves the performance. It is consistent with the argument of CMC (Tian et al., 2019a) that the more views, the better the representation, but our SelfCon learning is much more efficient in terms of the computational cost and GPU memory usage. However, for the efficiency of the experiments and a better understanding of the framework, we stuck to a single sub-network in all experimental settings.

We further experimented on the large-scale benchmarks, ImageNet-100 and ImageNet. In the ImageNet-100 experiment, we fixed the learning rate for SelfCon-M and SelfCon-S to 0.5, which was found to be optimal in the previous section. In Table 13, we found out that small sub-network showed the best performance in SelfCon-S as well as SelfCon-M. Therefore, we fixed the small sub-network with a learning rate of 0.5 for the SelfCon methods. For ImageNet dataset, we compared the results of small structure with those of same structure. The classification performance degraded by 0.3% when we used small structure. Therefore, we concluded that a deeper sub-network structure is preferred in the large-scale benchmark.

### C. Ablation Studies

#### C.1. Different Encoder Architectures

We experimented with other architectures: VGG-16 (Simonyan & Zisserman, 2014) with Batch Normalization (BN) (Ioffe & Szegedy, 2015) and WRN-16-8 (Zagoruyko & Komodakis, 2016), and the results are presented in Table 14. The classification accuracy for WRN-16-8 showed a similar trend as that of ResNet architectures. However, for VGG-16 with BN architecture, SupCon had a lower performance than CE on every dataset. Although the contrastive learning approach does not seem to result in significant changes for the VGG-16 with BN encoders, SelfCon-S was better than or comparable to CE.

#### C.2. ImageNet with Larger Batch Size

We summarized the full results of ImageNet benchmark in 3072 batch size in Table 15. SelfCon-M again showed the best result among others, and SelfCon-S still performed better than SupCon with only 63% of memory and 42% of time consumed. SelfCon-S had an ensemble performance of 73.2%, which is 2.0% higher than SupCon and 3.8% higher than CE. Meanwhile, SupCon-S failed to converge in this experiment. In the batch size of 3072, there are too many negative pairs without any positive feature from the same sample, such as an augmentation-based or feature-level multi-view. It makes the encoder difficult to find the positive pairs, and SupCon-S might fail to converge due to that reason. We tried a lower temperature value to sharpen the softmax vector or a different learning rate, but it also did not converge. Although SupCon-S does not work in a larger batch size, the SelfCon-S result implies that our framework is still compatible with the single-view.

### Table 14. The results of linear evaluation on WRN-16-8 and VGG-16 with BN for various datasets. We tuned the best structure and position of the sub-network for each architecture. Appendix B.4 summarizes the implementation details.

| Method          | Single-View   | WRN-16-8 | VGG-16 with BN          |
|-----------------|---------------|----------|-------------------------|
|                 |               | CIFAR-10 | CIFAR-100 | Tiny-ImageNet | CIFAR-10 | CIFAR-100 | Tiny-ImageNet |
| CE              | ✓             | 94.6±0.1 | 73.6±0.6 | 56.5±0.5 | 93.8±0.3 | 71.2±0.2 | 60.7±0.1 |
| SupCon          |               | 95.3±0.0 | 75.1±0.3 | 57.4±0.3 | 93.6±0.1 | 69.6±0.1 | 57.3±0.4 |
| SupCon-S        | ✓             | 95.2±0.1 | 76.0±0.1 | 57.3±0.5 | 93.8±0.3 | 71.1±0.0 | 58.4±0.2 |
| SelfCon-M (ours)|               | 95.4±0.2 | 75.6±0.1 | 58.7±0.1 | 93.4±0.1 | 71.7±0.3 | 59.4±0.1 |
| SelfCon-S (ours)| ✓             | 95.5±0.0 | 76.6±0.1 | 59.3±0.2 | 93.5±0.1 | 72.0±0.0 | 60.7±0.1 |

### Table 13. ImageNet-100 results on ResNet-18 with different sub-network structure.

| Method     | Sub-network Structure | Mem. Time. Acc@1 |
|------------|------------------------|------------------|
|            | FC | Small | Same |
| SelfCon-M  | 84.6 | 85.8 | 85.8 |
| SelfCon-S  | 85.5 | 86.3 | 85.7 |

### Table 15. The classification accuracy for ImageNet with 3072 batch size. We used a ResNet-18 architecture. X denotes the failure of the convergence.

| Method     | Mem. Time. Acc@1 |
|------------|------------------|
| CE         | - | - | 69.4 |
| SupCon     | ×0.9 | ×2.9 | 71.2 |
| SupCon-S   | ×1.0 | ×1.0 | 71.3 |
| SelfCon-M  | ×1.0 | ×1.0 | 71.7 |
| SelfCon-S  | ×1.0 | ×1.0 | 71.7 |
We formulate SelfCon loss in an unsupervised setting as SimCLR, using a positive set without label information. Although we have experimented only in supervision, our motivation of contrastive learning with a multi-exit framework can also be extended to unsupervised learning. We propose a SelfCon loss function for the unsupervised scenario and present the linear evaluation performance of ResNet-18 architecture on the CIFAR-100 dataset.

### C.3. Different Augmentation Policies

Multi-viewed methods have a problem that oracle needs to choose the augmentation policies carefully (Tian et al., 2019a; Chen et al., 2020; Caron et al., 2020; Kim et al., 2020). However, it is difficult and time-consuming to find the optimal policy. We investigated the following augmentation policies to claim that SelfCon-S reduces the burden of optimizing the augmentation policy.

- **Standard**: For standard augmentation, we used \{RandomResizedCrop, RandomHorizontalFlip, RandomColorJitter, RandomGrayscale\}. This is the same basic policy we used in the paper.

- **Simple**: When we do not have domain knowledge, it might be difficult to choose the appropriate augmentation policies. We assumed a scenario where we might not know that the color would be important in this visual recognition task. Therefore, we removed the color-related augmentation policies from Standard policy, i.e., we only used \{RandomResizedCrop, RandomHorizontalFlip\} for a simple augmentation policy.

- **RandAugment**: We used RandAugment (Cubuk et al., 2020) for an augmentation policy. RandAugment randomly samples \(N\) out of 14 transformation choices (e.g., shear, translate, autoContrast, and posterize) with \(M\) magnitude parameter. We used the optimized value of \(N = 2\) and \(M = 9\) in (Cubuk et al., 2020). It is already known that SupCon performs best with RandAugment (Khosla et al., 2020).

The results are presented in Table 16. When we apply Standard and Simple augmentations, SelfCon-S still outperformed SupCon. It supports that SelfCon learning is a more efficient algorithm because finding the best policy, such as RandAugment (Cubuk et al., 2020) or AutoAugment (Cubuk et al., 2019), is not a trivial process and needs a lot of computational costs. Meanwhile, SupCon with the multi-viewed batch can benefit more from the strong and optimized augmentation policy since training each sample twice more encourages memorization. SelfCon learning did not work well with RandAugment, as SupCon degraded with the Stacked RandAugment (Tian et al., 2020) in their experiments, but there would also be an optimal policy for SelfCon. We leave the experiments with other benchmarks, architectures, and various augmentation policies as future work.

### D. Extensions of SelfCon Learning

#### D.1. SelfCon in Unsupervised Learning

Although we have experimented only in supervision, our motivation of contrastive learning with a multi-exit framework can also be extended to unsupervised learning. We propose a SelfCon loss function for the unsupervised scenario and present the linear evaluation performance of ResNet-18 architecture on the CIFAR-100 dataset.

**Loss function** Under the unsupervised setting, Chen et al. (2020) proposed a simple framework for contrastive learning of visual representations (SimCLR) with NT-Xent loss. SimCLR suggests a contrastive task that contrasts the augmented pair of its own augmented image, i.e., \(P_i \equiv \{(i + B) \mod 2B\}\). We denote this loss as \(L_{\text{sim}}\).

We formulate SelfCon loss in an unsupervised setting as SimCLR, using a positive set without label information. We formulate SelfCon loss with the multi-viewed and unlabeled batch (SelfCon-MU) as follows:

\[
L_{\text{self-mu}} = \sum_{i \in I} \left[ \frac{1}{|P_i|} \sum_{p_1 \in P_i, \omega_1 \in \Omega} \omega(x_i)^T \omega_1(x_{p_1}) + \log \sum_{\omega_2 \in \Omega} \left( \frac{1}{|P_2|} \sum_{p_2 \in P_2} e^{\omega(x_i)^T \omega_2(x_{p_2})} + \frac{1}{|N_i|} \sum_{n \in N_i} e^{\omega(x_i)^T \omega_2(x_{n})} \right) \right]
\]

where

- \(i \in I \equiv \{1, \ldots, 2B\}\)
- \(p_j \in P_{ij} \equiv \{i, (i + B) \mod 2B\}\)
- \(n \in N_i \equiv I \setminus \{i, (i + B) \mod 2B\}\)

**Table 16. CIFAR-100 results on ResNet-18 with various augmentation policies.**

| Method       | Augmentation Policy |
|--------------|---------------------|
|              | Standard  | Simple   | RandAugment |
| SupCon       | 73.0 \pm 0.0 | 72.0 \pm 0.3 | 74.3 \pm 0.1 |
| SelfCon-S    | 75.4 \pm 0.1 | 74.2 \pm 0.2 | 72.5 \pm 0.0 |
Table 17. The results under the unsupervised scenario. We compared our SelfCon-MU and SelfCon-SU loss with SimCLR in the unsupervised setting. For the comparison with supervised learning, we also added the classification accuracy of CE loss. We used ResNet-18 encoder and CIFAR-100 dataset. Accuracy* denotes the accuracy of SelfCon learning with the anchors only from the sub-network (see details in Appendix D.2).

| Method         | CE       | SimCLR  | SelfCon-MU | SelfCon-SU |
|----------------|----------|---------|------------|------------|
| Multi-view     |          | ✓       | ✓          | ×          |
| Accuracy       | 72.9±0.1 | 63.3±0.3| 5.0±0.1    | 6.4±0.2    |
| Accuracy*      |          |         | 64.6±0.1   | 12.8±0.1   |

As SelfCon loss in supervised setting, we exclude an anchor sample from the positive set, i.e., \( P_{ij} \leftarrow P_{ij} \setminus \{i\} \) when \( \omega = \omega_j \). Here, we used \( \tau = 0.5 \) for the unsupervised SelfCon loss and \( \mathcal{L}_{\text{sim}} \). We omitted the dividing constant for the summation of anchor samples (i.e., \( \frac{1}{|I|} \)).

We also formulate SelfCon loss with the single-viewed and unlabeled batch (SelfCon-SU) as follows:

\[
\mathcal{L}_{\text{self-su}} = \sum_{\omega \in \Omega} \left[ -\frac{1}{|P_{1i}|} \sum_{p_1 \in P_{1i}, \omega_1 \in \Omega} \omega(x_i)^\top \omega_1(x_{p_1}) + \log \sum_{\omega_2 \in \Omega} \left( \sum_{p_2 \in P_{2i}} e^{\omega(x_i)^\top \omega_2(x_{p_2})} + \sum_{n \in N_i} e^{\omega(x_i)^\top \omega_2(x_n)} \right) \right]
\]

(43)

Similarly, we exclude the sample from the positive set only when \( \omega = \omega_j \), i.e., \( P_{ij} \leftarrow P_{ij} \setminus \{i\} \). For the positive set, since this loss is based on the single-viewed batch, we have an empty positive set when \( \omega = \omega_j \).

**Experimental results** All implementation details for unsupervised representation learning are identical with those of supervised representation learning in Appendix B, except for temperature \( \tau \) of 0.5 and linear evaluation learning rate of 1.0. We used a small sub-network attached after the 2nd block. Table 17 shows the linear evaluation performance of unsupervised learning on ResNet-18 in CIFAR-100 dataset. However, we empirically found that the encoder failed to converge with both SelfCon-MU and SelfCon-SU loss (see the accuracy of the first row).

### D.2. SelfCon with Anchors ONLY from the Sub-network

We suspect that Eq. 42 and 43 allow the backbone network to follow the sub-network, which makes the last feature learn more redundant information about the input variable without any label information. Thus, the unsupervised loss function under the SelfCon framework needs to be modified.

When the anchor feature is from the backbone network, we remove the loss term, which contrasts the features of the sub-network. Strictly speaking, it does not perfectly prevent the backbone from following the sub-network because there is no stop-gradient operation on the outputs of the backbone network when the outputs of the sub-network are the anchors. However, we hypothesize that it helps prevent the encoder from collapsing to the trivial solution by the contradiction of the IB principle. We confirmed the performance of revised loss functions in both unsupervised and supervised scenarios.

**Loss function**

\[
\mathcal{L}_{\text{self-mu}}^* = \frac{1}{1 + \alpha} \mathcal{L}_{\text{sim}}
\]

\[
+ \frac{\alpha}{1 + \alpha} \sum_{i \in I} \left[ -\frac{1}{|P_{1i}|} \sum_{p_1 \in P_{1i}, \omega_1 \in \Omega} G(x_i)^\top G(x_{p_1}) + \log \sum_{\omega_2 \in \Omega} \left( \sum_{p_2 \in P_{2i}} e^{G(x_i)^\top \omega_2(x_{p_2})} + \sum_{n \in N_i} e^{G(x_i)^\top \omega_2(x_n)} \right) \right]
\]

(44)

All notations are same as Eq. 42, except for the coefficient \( \alpha \) where we used 1.0. For the supervised setting, simply change \( P_{ij} \) to \( \{p_j \in I | y_p = y_i\} \) and \( \mathcal{L}_{\text{sim}} \) to \( \mathcal{L}_{\text{sup}} \). Note that \( P_{ij} \leftarrow P_{ij} \setminus \{i\} \) when \( \omega_j = G \). We get rid of the situation that the
Table 18. CIFAR-100 results with SelfCon extensions. Accuracy* denotes the accuracy of SelfCon learning with the anchors only from the sub-network. We used $\alpha = 1$ and a fully-connected layer as the sub-network structure.

| Method  | Architecture | Accuracy | Accuracy* |
|---------|--------------|----------|-----------|
| SelfCon-M | ResNet-18     | 74.9 ±0.1 | 74.9 ±0.2 |
| SelfCon-S |             | 75.4 ±0.1 | 75.6 ±0.1 |
| SelfCon-M | ResNet-50     | 76.9 ±0.1 | 77.4 ±0.2 |
| SelfCon-S |             | 78.5 ±0.3 | 78.8 ±0.1 |

Table 19. The detailed results of Figure 2. $x$, $y$, $T(x)$, and $F(x)$ respectively denote the input variable, label variable, intermediate feature, and the last feature. Recall that $T(x)$ is the intermediate feature of the backbone network, which is an input to the auxiliary network path. We summarized the average of estimated MI through multiple random seeds. We highlighted the MI between the intermediate and the last features, which is the main concern of the SelfCon loss. **Bold** type indicates the smallest values for $\mathcal{I}(x; T(x))$ and the largest values for $\mathcal{I}(y; T(x))$ and $\mathcal{I}(F(x); T(x))$, according to the IB principle. We used ResNet-18 on CIFAR-100 dataset for the measurements.

| Method  | CE | SupCon | SelfCon-M* | SelfCon-M* | SelfCon-M* | SelfCon-M* | SelfCon-M* | SelfCon-M* | SelfCon-M* |
|---------|----|--------|------------|------------|------------|------------|------------|------------|------------|
|         |    |        |            |            |            |            |            |            |            |
|         | 72.9 | 73.0 | 0.025 | 0.05 | 0.075 | 0.1 | 0.15 | - | - |
| InfoNCE | $\mathcal{I}(x; T(x))$ | 0.436 | 0.285 | 0.296 | 0.277 | 0.299 | 0.293 | 0.232 | **0.203** | **0.207** |
|         | $\mathcal{I}(y; T(x))$ | 0.233 | 0.221 | 0.299 | 0.290 | 0.309 | 0.329 | 0.341 | 0.454 | **0.463** |
| MINE    | $\mathcal{I}(x; F(x); T(x))$ | 0.313 | 0.296 | 0.330 | 0.357 | 0.381 | 0.392 | 0.402 | 0.508 | **0.528** |
| NWJ     | $\mathcal{I}(x; F(x); T(x))$ | 1.225 | 0.758 | 0.843 | 0.719 | 0.744 | 0.665 | 0.697 | 0.508 | **0.503** |
|         | $\mathcal{I}(y; F(x))$ | 0.617 | 0.616 | 0.719 | 0.700 | 0.834 | 0.928 | 0.961 | 1.261 | **1.425** |

For the supervised setting, we change the above equation as the equally-weighted linear combination of SupCon-S loss and Eq. 45 with $F_{ij} = \{p_j \in I | y_p = y_i\}$. Note that we also exclude contrasting the anchor itself in SupCon-S loss term.

**Experimental results** In Table 17, we also reported the accuracy of SelfCon-MU* and SelfCon-SU* loss according to Eq. 44 and 45. Surprisingly, in this case, SelfCon-MU* outperformed SimCLR loss (Chen et al., 2020), improving 1.3%p. Unfortunately, SelfCon-SU* had not converged again, although it improved the result in a small amount compared to Eq. 43. While SelfCon-MU* has SimCLR loss term that makes the backbone encoder still learn meaningful features, SelfCon-SU* loss does not have the anchor features from the backbone, which makes the backbone hard to be trained. Table 18 summarizes SelfCon-M* and SelfCon-S* loss, removing the anchors from the backbone in the supervised setting, i.e., supervised version of Eq. 44 and 45. As we expected, these variants of SelfCon-M and SelfCon-S further improved the classification performance.

**E. Correlation Between SelfCon Loss and the MI Estimation**

We used three types of MI estimators: InfoNCE (Oord et al., 2018), MINE (Belghazi et al., 2018), and NWJ (Nguyen et al., 2010). Specifically, we extracted the features of the CIFAR-100 dataset from the pretrained ResNet-18 encoders and optimized a simple 3-layer Conv-ReLU network with the MI estimator objectives.

In Section 5.2, to clearly show the correlation between the mutual information and classification accuracy, we experimented...
Table 20. **Memory (GiB / GPU) and computation time (sec / step) comparison.** All numbers are measured with ResNet-50. Note that FLOPS is for one sample. $B$ stands for batch size.

| Dataset (Image size) | Method       | Params  | FLOPS   | $B = 256$ | $B = 512$ | $B = 1024$ |
|----------------------|--------------|---------|---------|-----------|-----------|------------|
|                      |              |         | Memory  | Time      | Memory    | Time       |
| CIFAR-100 (32x32)    | SupCon       | 27.96 M | 2.62 G  | 4.00      | 0.28      | 6.40       | 0.35       | 11.29      | 0.50      |
|                      | SelfCon-S    | 33.47 M | 1.31 G  | 2.73      | 0.28      | 3.92       | 0.31       | 6.28       | 0.40      |
| Tiny-ImageNet (64x64)| SupCon       | 27.96 M | 2.63 G  | 4.41      | 0.27      | 6.71       | 0.33       | 11.84      | 0.46      |
|                      | SelfCon-S    | 33.47 M | 1.32 G  | 2.98      | 0.27      | 4.21       | 0.29       | 6.82       | 0.34      |
| ImageNet-100 (224x224)| SupCon   | 27.97 M | 8.28 G  | 9.41      | 0.61      | 16.49      | 1.14       | -          | -         |
|                      | SelfCon-S    | 42.21 M | 5.33 G  | 5.91      | 0.47      | 10.45      | 0.72       | -          | -         |

with the interpolation between SupCon loss and SelfCon-M loss (SupCon loss is a special case of SelfCon-M loss). However, the current formulation of Eq. 1 and Eq. 2 cannot make the exact interpolation between SupCon and SelfCon-M because the SelfCon-M loss should have negative pairs from different levels of a network (i.e., backbone and sub-network), but the SupCon loss cannot produce those. Therefore, we proposed a near-interpolated loss function between SupCon and SelfCon-M loss, which is equivalent to the supervised version of Eq. 44:

$$
\mathcal{L}_{self-m*} = \frac{1}{1 + \alpha} \mathcal{L}_{sup}
+ \frac{\alpha}{1 + \alpha} \sum_{i \in I} \left[ \sum_{p_1 \in P_1, \omega_1 \in \Omega} G(x_i)^\top \omega_1 (x_{p_1}) + \log \sum_{\omega_2 \in \Omega} \left( \sum_{p_2 \in P_2} e^{G(x_i)^\top \omega_2 (x_{p_2})} + \sum_{n \in N_i} e^{G(x_i)^\top \omega_2 (x_n)} \right) \right]
$$

(46)

where $P_{ij} \leftarrow P_{ij} \setminus \{i\}$ when $\omega_j = G$. Therefore, if $\alpha = 0$, $\mathcal{L}_{self-m*}$ is equivalent to the SupCon loss and if $\alpha = 1$, $\mathcal{L}_{self-m*}$ is almost equivalent to SelfCon-M loss.

Figure 2 describes the estimated mutual information and its relationship with classification performance via controlling the hyperparameter $\alpha$, and Table 19 summarizes the detailed estimation values of the intermediate feature with respect to the input, label, and the last feature. As expected, SelfCon-M and SelfCon-S loss have larger MI between the intermediate and the last feature of the backbone network than CE and SupCon loss. We observed a clear increasing trend of both MI and test accuracy as the contribution of SelfCon gets larger (i.e., increasing $\alpha$). When we used a fully-connected layer as the sub-network, we confirmed that the accuracy of SelfCon-M$^*$ was quickly saturated to SelfCon-M for $\alpha = 512$. In other words, SelfCon-S with $\alpha = 512$ requires a comparable memory cost with SupCon with $\alpha = 256$.

F. Memory Usage and Computational Cost

In Table 20, we reported the computational cost for pretraining with ResNet-50. The 1024 batch size results on the ImageNet-100 benchmark are not reported because ResNet-50 with batch size over 1024 exceeded the GPU limit. All numbers for the ImageNet-100 and ImageNet results are measured on 8 RTX A5000 GPUs, and in other benchmarks on 8 RTX 2080 Ti GPUs. The overall trend is similar to that in Table 5. Although SelfCon-S has larger parameters due to the auxiliary networks, the actual training memory and time were lower than SupCon because of the single-viewed batch. In other words, SelfCon-S with $B = 512$ requires a comparable memory cost with SupCon with $B = 256$.

G. Ensemble Performance of SelfCon-S in ResNet-50

We pointed out the ensemble as the advantage of the SelfCon learning framework. Ensembling the multiple outputs is still efficient because the additional fine-tuning of a linear classifier after the frozen sub-network does not place a lot of burden on the computational cost. We summarized the experimental results for ResNet-18 in Table 7 and ResNet-50 in Table 21. Ensemble improved the performance by a large margin in every encoder architecture and benchmark. In addition, the ensemble results of ResNet-18 significantly outperformed SupCon on ResNet-50: SelfCon-S on ResNet-18 77.4%
Table 21. **Classification accuracy with the classifiers after backbone, sub-network, and the ensemble of them.** The ResNet-50 encoder is pretrained by the SelfCon-S loss function.

| Method     | CIFAR-10 | CIFAR-100 | Tiny-ImageNet | ImageNet-100 |
|------------|----------|-----------|---------------|--------------|
| SupCon     | 95.6     | 75.5      | 61.6          | 88.2         |
| Backbone   | **95.7** | 78.5      | 63.7          | 88.5         |
| Sub-network| 93.6     | 73.3      | 58.9          | 87.4         |
| Ensemble   | 95.5     | **80.0**  | **65.7**      | **89.0**     |

vs. SupCon on ResNet-50 75.5% (CIFAR-100) and SelfCon-S on ResNet-18 62.2% vs. SupCon on ResNet-50 61.6% (Tiny-ImageNet).

**H. Qualitative Examples for Vanishing Gradient**

In Figure 6, we have already shown that the sub-network solves the vanishing gradient problem through the visualization for gradient norms of each layer. In Figure 7, we also visualized qualitative examples using Grad-CAM (Selvaraju et al., 2017). We used the gradient measured on the last layer in the 2nd block when the sub-network is attached after the 2nd block. In order to compare the absolute magnitude of the gradient, it is normalized by the maximum and minimum values of the two methods, SelfCon-M and SupCon. As in Figure 7, SelfCon learning led to a larger gradient via sub-networks, and Grad-CAM more clearly highlighted the pixels containing important information in the images.