MetAug: Contrastive Learning via Meta Feature Augmentation

Jiangmeng Li 1 2 3 Wenwen Qiang 1 2 3 Changwen Zheng 1 3 Bing Su 4 5 Hui Xiong 6 7

Abstract
What matters for contrastive learning? We argue that contrastive learning heavily relies on informative features, or "hard" (positive or negative) features. Early works include more informative features by applying complex data augmentations and large batch size or memory bank, and recent works design elaborate sampling approaches to explore informative features. The key challenge toward exploring such features is that the source multi-view data is generated by applying random data augmentations, making it infeasible to always add useful information in the augmented data. Consequently, the informativeness of features learned from such augmented data is limited. In response, we propose to directly augment the features in latent space, thereby learning discriminative representations without a large amount of input data. We perform a meta learning technique to build the augmentation generator that updates its network parameters by considering the performance of the encoder. However, insufficient input data may lead the encoder to learn collapsed features and therefore malfunction the augmentation generator. A new margin-injected regularization is further added in the objective function to avoid the encoder learning a degenerate mapping. To contrast all features in one gradient back-propagation step, we adopt the proposed optimization-driven unified contrastive loss instead of the conventional contrastive loss. Empirically, our method achieves state-of-the-art results on several benchmark datasets.

1. Introduction
Contrastive learning methods have achieved empirical success in computer vision (Chopra et al., 2005; Hadsell et al., 2006). Under the setting of self-supervised learning (SSL), recent researches demonstrate the superiority of contrastive methods (Hjelm et al., 2018; Tian et al., 2019; Chuang et al., 2020; Robinson et al., 2020). Typically, these approaches learn features by contrasting different views (e.g., different random data augmentations) of the image in hidden space. We recap the preliminaries of the conventional contrastive learning paradigm: every two views of the same image are considered to be a positive pair, and every two views of the different images are considered to be a negative pair; the contrastive loss (Gutmann & Hyvärinen, 2010; Oord et al., 2018) guides the learned features to bring positive pairs together and push negative pairs farther apart.

However, this learning paradigm suffers from the need for a large number of pairs to contrast, e.g., large batch size or memory bank size, because many pairs are not informative to the model, i.e., positive pairs are pretty close and negative pairs are already very apart in hidden space. These pairs have few contributions to the optimization. Contrastive methods need numerous pairs and expect to collect informative ones, and therefore complex data augmentations (e.g., jittering, random cropping, separating color channels, etc.) (Bachman et al., 2019; Chen et al., 2020) and large-scale memory banks (Tian et al., 2019; He et al., 2020) are effective in improving the performance of contrastive models on downstream tasks.

The success of the recent works depends on the elaborate selection of informative negative pairs (Chuang et al., 2020; Robinson et al., 2020). These methods focus on designing sampling strategies to assign larger weights to informative pairs, which rely on enough and informative positive pairs and do not need large amounts of negative pairs. When the number of pairs to contrast is limited, the contrastive loss may cause conventional contrastive learning approaches to learn collapsed features (Zbontar et al., 2021; Grill et al.,

---

1 Equal contribution 1 Science & Technology on Integrated Information System Laboratory, Institute of Software Chinese Academy of Sciences, Beijing, China 2 University of Chinese Academy of Sciences, Beijing, China 3 Southern Marine Science and Engineering Guangdong Laboratory (Guangzhou), Guangdong, China. 4 Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China 5 Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing, China 6 Thrust of Artificial Intelligence, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China 7 Department of Computer Science Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China. Correspondence to: Bing Su <subingats@gmail.com>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).
Nowadays, many researchers have noticed potential environmental problems brought by training deep learning models (Xu et al., 2021), for instance, (Strubell et al., 2019) reports a remarkable example that the carbon dioxide emissions generated by training a Transformer (Vaswani et al., 2017) is equivalent to 200 round trips between San Francisco and New York by plane. Therefore, we motivate our method to perform an efficient self-supervised contrastive approach to learn anti-collapse and discriminative features based on a restricted amount of images in a training epoch (e.g., small batch size) and plain neural networks with limited parameters. Much research effort has been devoted to strong augmentations on data, but the informativeness of the features learned from the augmented data is hard to exactly measure, since the data is fed into mapping-agnostic deep neural networks to generate the features. Instead, we directly tackle augmentations on features and show that appropriate feature augmentations can sharply improve the optimization.

To this end, we propose Meta Feature Augmentation (MetAug), which learns view-specific encoders (with projection heads) and auxiliary meta feature augmentation generators (MAGs) by margin-injected meta feature augmentation and optimization-driven unified contrast. Suppose the input data has $M$ views, and the multi-view data is fed into the encoder to generate the latent features. We initialize $M$ neural networks as MAGs for views, which are used to augment the features of each view. We contrast all original and augmented features for bi-optimization training. Through such a learning paradigm, MetAug can improve the performance of self-supervised contrastive learning.

To learn anti-collapse and discriminative features from a restricted amount of images, MetAug relies on two key ingredients: 1) margin-injected meta feature augmentation, where MAGs use the performance of the encoder in one iteration to improve the view-specific feature augmentations for the next iteration. In this way, MAGs promote the encoder to efficiently explore the discriminative information of the input. For the original features and the augmented features generated by MAGs, we inject a margin $R_\sigma$ between the similarities of them, which avoids the instance-level feature collapse; 2) optimization-driven unified contrast, which contrasts all features in one gradient back-propagation step. Such proposed contrast can also amplify the impact of the instance similarity that deviates far from the optimum and weaken the impact of the instance similarity that is close to the optimum. We conduct head-to-head comparisons on various benchmark datasets, which prove the effectiveness of margin-injected meta feature augmentation and optimization-driven unified contrast. Contributions:

- We propose margin-injected meta feature augmentation, which directly augments the latent features to generate informative and anti-collapse features. Benefiting from such features, encoders can efficiently capture discriminative information.
- We propose optimization-driven unified contrast to include all available features in one step of back-propagation and weight the similarities of paired features by measuring their contributions to optimization.
- Empirically, MetAug improves the downstream task performance on different benchmark datasets.

2. Related works

Self-supervised learning. Under the setting of unsupervised learning, SSL methods have achieved impressive success, which constructs auxiliary tasks to learn discriminative information from the unlabeled inputs. Deep InfoMax (Hjelm et al., 2018) explores to maximize the mutual information between an input and the output of a deep neural network encoder by different mutual information estimations. CPC (Oord et al., 2018) proposes to adopt noise-contrastive estimation (NCE) (Gutmann & Hyvärinen, 2010) as the contrastive loss to train the model to measure the mutual information of multiple views deduced by the Kullback-Leibler divergence (Goldberger et al., 2003). CMC (Tian et al., 2019) and AMDIM (Bachman et al., 2019) employ contrastive learning on the multi-view data. SwAV (Caron et al., 2020) compares the cluster assignments under different views instead of directly comparing features by using more views (e.g., six views). SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) use large batch or memory bank to enlarge the amount of available negative features to learn good representations. Instead of exploring informative features by adopting various data augmentations and enlarging the number of features, our method focuses on straightforwardly generating informative features to contrast.

Recent works explore imposing stronger constraints on the conventional contrastive learning paradigm or propose alternative loss functions (instead of contrastive loss). DebiasedCL (Chuang et al., 2020) and HardCL (Robinson et al., 2020) consider to directly collect informative features to contrast by designing sampling strategies, which are inspired by positive-unlabeled learning methods (Elkan & Noto, 2008; du Plessis et al., 2014). Motivated by (Sridharan & Kakade, 2008), (Tsai et al., 2020) proposes an information theoretical framework for SSL, which, guided by the theory, uses information bottleneck to restrict the learned features and maintain the sufficient self-supervision. BYOL (Grill et al., 2020), W-MSE (Ermonov et al., 2020), and Barlow Twins (Zbontar et al., 2021) present a crucial issue that insufficient self-supervision (e.g., not enough negative features) may lead to the feature collapse in hidden space. To tackle the
mentioned issue, we propose a new margin-injected regularization in meta feature augmentation to avoid generating degenerate features. DACL (Verma et al., 2021) proposes a new data augmentation that applies to domain-agnostic problems. LooC (Xiao et al., 2021) learns to capture varying and invariant factors for visual representations by constructing separate embedding spaces for each augmentation. These methods explore informative features from the perspective of data augmentation, while our straightforward idea behind our method is to augment features in the latent space.

**Meta learning.** The objective of meta learning is to automatically learn the meta learning algorithm. Early works (A et al., 2002; Bengio et al., 2002; Schmidhuber, 2014) aim to guide the model (e.g., neural network) to learn prior knowledge about how to learn new knowledge, so that the model can efficiently learn new knowledge, e.g., the model can be quickly fine-tuned to specific downstream tasks with few training steps and achieve good performance. Recently, researchers explored to use meta learning to find optimal hyper-parameters (Li et al., 2017) and appropriately initialize a neural network for few-shot learning (Finn et al., 2017; Snell et al., 2017; Vinyals et al., 2016). Recent approaches (Chen et al., 2016; Jaderberg et al., 2016; Ma et al., 2018; Liu et al., 2019) have focused on learning optimizers or generating a gradient-driven loss for deep neural networks in the field of NLP, computer vision, etc.

### 3. Method

Our goal is to learn representations that capture information shared between multiple different views by performing self-supervised contrastive learning. Formally, we suppose the input multi-view dataset as $X = \{X_1, X_2, ..., X_N\}$, where $N$ denotes the number of samples. $X_i$ represents a collection of $M$ views of the sample, where $i \in \{1, ..., N\}$. For each sample $X_i$, we denote $x_i$ as a random variable representing views following $x_i \sim \mathcal{P}(X_i)$, and $x_i^j$ denotes the $j$-th view of the $i$-th sample, where $j \in \{1, ..., M\}$.

#### 3.1. Contrastive learning preliminary

We recap the preliminaries of contrastive learning(Tian et al., 2019; Chen et al., 2020): the foundational idea behind contrastive learning is to learn an embedding that maximizes agreement between the views of the same sample and separates the views of different samples in latent space. Given a multi-view dataset $X$, we treat pairs of the views of the same sample $\{x_i^j, x_i^{j'}\}$, where $j, j' \in \{1, ..., M\}$, as positives, versus pairs of the views of the different samples $\{x_i^j, x_i'^j\}$, where $i \neq i'$, as negatives. To impose contrastive learning, we need the input $x_i^j$ into a view-specific encoder $f_{\theta_j}(\cdot)$ to learn a representation $h_i^j$, and $h_i^j$ is mapped into a feature $z_i^j$ by a projection head $g_{\vartheta_j}(\cdot)$, where $\theta_j$ and $\vartheta_j$ are the network parameters of $f_{\theta_j}(\cdot)$ and $g_{\vartheta_j}(\cdot)$, respectively. A discriminating function $d(\cdot)$ is adopted to measure the similarity of $\{z_i^j, z_i'^j\}$, where $i \neq i'$. The encoder $f_{\theta_j}(\cdot)$ and projection head $g_{\vartheta_j}(\cdot)$ are trained by using a contrastive loss (Oord et al., 2018), which is formulated as follows:

$$
\mathcal{L} = - \mathbb{E}_{X_S} \log \frac{d(\{z^+\})}{d(\{z^+\}) + \sum_{k=1}^{K} d(\{z^-\})}
$$

(1)

where $X_S = \{\{z^+\}, \{z^-\}_1, \{z^-\}_2, ..., \{z^-\}_K\}$ is a set of pairs randomly sampled from $X$, which includes a positive $\{z^+\}$ and $K$ negatives $\{z^-\}_k$, $k \in \{1, ..., K\}$, because contrastive loss can only use one positive in an iteration. In test, the projection head $g_{\vartheta_j}(\cdot)$ is discarded, and the representation $h_i^j$ is directly used for downstream tasks.

#### 3.2. Margin-injected meta feature augmentation

Recent contrastive methods rely on complex data augmentations to increase the informativeness of views. Yet this lack of guidance approach leads to the demand for a large number of training data (e.g., large batch size and memory bank). We propose a meta feature augmentation method,
which creates informative augmented features by updating parameters of its own network according to the performance (gradient) of the encoder (see Appendix A.3 for our rethinking of augmented features). A visualization of the overall MetAug architecture is shown in Figure 1.

To this end, we build a group of MAGs \( a_{\omega}() = \{a_{\omega 1}(), ..., a_{\omega M}()\} \) for all \( M \) views, where \( \omega = \{\omega 1, ..., \omega M\} \). To be simplified, we define \( f_{\theta} \) and \( g_{\theta} \) as the groups of view-specific encoders and projection heads, respectively, i.e., \( \theta = \{\theta 1, ..., \theta M\} \) and \( \vartheta = \{\vartheta 1, ..., \vartheta M\} \).

In training, the encoders \( f_{\theta}() \) and the projection heads \( g_{\theta}() \) are trained alongside the MAG \( a_{\omega}() \) (with the network parameters \( \omega \)). Following the protocol of meta learning (Finn et al., 2017; Liu et al., 2019), we firstly train \( f_{\theta}() \) and \( g_{\theta}() \) under the learning paradigm of self-supervised contrastive learning. Then, \( a_{\omega}() \) is updated by computing its gradients with respect to the performance of \( f_{\theta}() \) and \( g_{\theta}() \). Here, we measure the performance of \( f_{\theta}() \) and \( g_{\theta}() \) by the gradients of them when the corresponding contrastive loss is back-propagated. Concretely, all of \( f_{\theta}(), g_{\theta}(), \) and \( a_{\omega}() \) are iteratively trained until convergence.

Specifically, we first update network parameters \( \theta \) and \( \vartheta \) of the encoders and projection heads by adopting the conventional contrastive loss. Then, we train the MAG \( a_{\omega}() \) in a meta learning manner. We encourage the augmented features to be informative, and the encoders \( f_{\theta}() \) can better explore the discriminative information by jointly using the original and augmented features to contrast. Hence, the performance of the encoders would be promoted on the same training data. To update network parameters \( \omega \) of the \( a_{\omega}() \), we formalize the meta updating objective as follows:

\[
\arg \min_{\omega} \mathcal{L} \left( \{g_{\theta}(f_{\theta}(\bar{X})), a_{\omega}(g_{\theta}(f_{\theta}(\bar{X})))\} \right)
\]

(2)

where \( \bar{X} \) represents a minibatch sampled from the training dataset \( X \), \( \{g_{\theta}(f_{\theta}(\bar{X})), a_{\omega}(g_{\theta}(f_{\theta}(\bar{X})))\} \) denotes a set including both original features and meta augmented features. \( \vartheta \) and \( \theta \) represent the parameter sets of the encoders and projection heads, respectively, which are computed by the updating of one gradient back-propagation:

\[
\hat{\theta} = \theta - \ell \cdot \nabla_{\theta} \mathcal{L} \left( \{g_{\theta}(f_{\theta}(\bar{X})), a_{\omega}(g_{\theta}(f_{\theta}(\bar{X})))\} \right)
\]

\[
\hat{\vartheta} = \vartheta - \ell \cdot \nabla_{\vartheta} \mathcal{L} \left( \{g_{\theta}(f_{\theta}(\bar{X})), a_{\omega}(g_{\theta}(f_{\theta}(\bar{X})))\} \right)
\]

(3)

where \( \ell \) is the learning rate shared between \( \theta \) and \( \vartheta \). The idea behind the meta updating objective is that we perform the second-derivative technique (Finn et al., 2017; Zhang et al., 2018; Liu et al., 2019) to train \( a_{\omega}() \). Specifically, a derivative over the derivative (Hessian matrix) of the combination \( \{\theta, \vartheta\} \) is used to update \( \omega \), where \( \{\theta, \vartheta\} \) is a parameter set conjoining \( \theta \) and \( \vartheta \). We compute the derivative with respect to \( \omega \) by using a retained computational graph of \( \{\theta, \vartheta\} \).

However, in practice, we find a critical issue: when the original features are not informative enough, large gradients are difficult to generate by contrasting the uninformative features, the MAGs \( a_{\omega}() \) are inclined to create collapsed augmented features, e.g., the augmented features and the original features are very similar. We consider the reason for the feature collapse is that small gradient changes of the encoders alongside projection heads \( g_{\theta}(f_{\theta}()) \) lead to the update step-size of \( a_{\omega}() \) to become extensively small, which leaves the optimization of \( a_{\omega}() \) to fall into a local optimum. The augmented features are such that without any extra useful information. To tackle this issue, we further inject a margin to encourage \( a_{\omega}() \) to generate more complex and informative augmented features, which can be considered as a regularization term in the meta updating objective. See Figure 2(a) for the details of the augmented feature collapse issue, and we observe that, without margin-injected regularization, MAGs tend to generate collapsed features that are very similar with the original features. Formally, we formulate the approach to generate margins for \( a_{\omega}() \) by

\[
\sigma^+ = \min \left[ \min \left\{d(\{z^+\}_{k+})\right\}, \max \left\{d(\{z^-\}_{k-})\right\} \right]
\]

\[
\sigma^- = \max \left[ \min \left\{d(\{z^+\}_{k+})\right\}, \max \left\{d(\{z^-\}_{k-})\right\} \right]
\]

(4)

where \( \{d(\{z^+\}_{k+})\} \) is a set of the outputs (similarities)
of positives computed by the discriminating function \( d(\cdot) \), and \( k^+ \in \{1, \ldots, K^+\} \) where \( K^+ \) represents the number of positives in a minibatch. \( \{d(\{z\}^+_{k^-})\} \) is a set of the discriminating outputs of negatives, and \( k^- \in \{1, \ldots, K^\} \) where \( K^- \) represents the number of negatives. Note that only original features are used in Equation 4. We call the formulated margin generation approach as "Large"; and we also propose two more approaches, called "Medium" and "Small". In Appendix A.2, we conduct comparisons to evaluate the effects of the three margin generation approaches. We inject the margins between the augmented features and original features by adding a regularization term in the meta updating objective, and the regularization is defined as:

\[
\mathcal{R}_\sigma = \frac{1}{K^+} \sum_{k^+ = 1}^{K^+} \left[ d(\{\hat{z}^+\}_{k^+}) - \sigma^+ \right] + \frac{1}{K^-} \sum_{k^- = 1}^{K^-} \left[ \sigma^- - d(\{\hat{z}^-\}_{k^-}) \right] +
\]

where \( \{\hat{z}^+\}_{k^+} \) denotes a positive that includes one original feature and one augmented feature, and \( K^+ \) denotes the number of such positives. \( \{\hat{z}^-\}_{k^-} \) represents likewise one of \( K^- \) negatives, each of which includes one original feature and one augmented feature. \([\cdot]^+\) denotes the cut-off-at-zero function, which is defined as \( [a]^+ = \max(a, 0) \). We then integrate such regularization to the updating of \( \omega \) by

\[
\omega \leftarrow \omega - \ell' \cdot \nabla \omega \ell \left( \{g_\theta(f_\theta(\bar{X}))\}, a_\omega \left( \{g_\theta(f_\theta(\bar{X}))\} \right) \right) + \alpha \cdot \mathcal{R}_\sigma
\]

where \( \ell' \) represents the learning rate of \( \omega \), and \( \alpha \) is a hyperparameter balancing the impact of the loss of margin-injected regularization term. \( \mathcal{R}_\sigma \) restricts MAGs to generate informative features that are more different with the original features (see Figure 2(b)). In practice, Figure 2(c) and (d) show that the features learned by our method (with margin-injected regularization) are more concentrated, e.g., the features of the same image are more similar and the gap between the features of the different images are enlarged, which proves informative augmented features can further lead the encoders to learn non-collapsed (scattered) features.

### 3.3. Optimization-driven unified contrast

We propose to jointly contrast all features (including the original features and the meta augmented features) in one gradient back-propagation step. Motivated by (Schroff et al., 2015), we introduce the following optimization-driven unified loss function to replace the conventional contrastive loss as follows:

\[
\mathcal{L}_{OUCL} = \left[ \sum_{k^- = 1}^{K^-} d(\{z^-\}_{k^-}) - \sum_{k^+ = 1}^{K^+} d(\{z^+\}_{k^+}) + \lambda \right] +
\]

where \([\cdot]^+\) ensures that \( \mathcal{L} \geq 0 \) is always held. Note that all original features and augmented features are involved. \( \lambda \) is a margin between the summarized instance similarities to enhance the capability of the similarity separation. However, we find that the difference between \( \sum_{k^- = 1}^{K^-} d(\{z^-\}_{k^-}) \) and \( \sum_{k^+ = 1}^{K^+} d(\{z^+\}_{k^+}) \) is not the larger the better. Excessive increases of the difference may undermine the convergence in optimization. We thereby wish to adopt a margin \( \lambda \) that leads to preferable convergence. We reform the loss in Equation 7 by adding a temperature coefficient \( \beta \) as follows:

\[
\mathcal{L}_{OUCL} = \frac{1}{\beta} \log \left\{ 1 + \sum_{k^- = 1}^{K^-} \sum_{k^+ = 1}^{K^+} \exp \left[ \beta \left( d(\{z^-\}_{k^-}) - d(\{z^+\}_{k^+}) + \lambda \right) \right] \right\}
\]

when \( \beta \to +\infty \), Equation 8 is Equation 7. Inspired by (Sun et al., 2020), we use weighting factors \( \Gamma^- \) and \( \Gamma^+ \) to modulate the impacts of \( d(\{z^-\}_{k^-}) \) and \( d(\{z^+\}_{k^+}) \). Such approach aims to give greater weight to the similarity.

---

**Algorithm 1 MetAug**

**Input:** Multi-view dataset \( \bar{X} \) with \( M \) views of each sample, minibatch size \( n \), and hyperparameters \( \alpha, \beta, \delta \).

**Initialize** The neural network parameters: \( \theta \) and \( \vartheta \) for view-specific encoders \( f_\theta(\cdot) \) and projection heads \( g_\theta(\cdot) \), \( \omega \) for MAGs, i.e., \( a_\omega(\cdot) \). The learning rates: \( \ell \) and \( \ell' \).

repeat for \( t \)-th training iteration do

- Iteratively sample minibatch \( \bar{X} = \{X_i\}_{i=(t-1)n}^{tn} \).
- \# regular contrastive training step
- \( \theta \leftarrow \theta - \ell \Delta_\theta \mathcal{L}_{MetAug}(f_\theta, g_\theta, a_\omega, \bar{X}) \)
- \( \vartheta \leftarrow \vartheta - \ell \Delta_\vartheta \mathcal{L}_{MetAug}(f_\theta, g_\theta, a_\omega, \bar{X}) \)

end for

for \( t \)-th training iteration do

- Iteratively sample minibatch \( \bar{X} = \{X_i\}_{i=(t-1)n}^{tn} \).
- \# compute fast weights
- \( \theta = \theta - \ell \Delta_\theta \mathcal{L}_{MetAug}(f_\theta, g_\theta, a_\omega, \bar{X}) \)
- \( \vartheta = \vartheta - \ell \Delta_\vartheta \mathcal{L}_{MetAug}(f_\theta, g_\theta, a_\omega, \bar{X}) \)
- \# meta training step using second derivative
- \( \omega \leftarrow \omega - \ell' \Delta_\omega \left( \mathcal{L}_{MetAug}(f_\theta, g_\theta, a_\omega, \bar{X}) + \alpha \cdot \mathcal{R}_\sigma \right) \)

end for

until \( \theta, \vartheta, \) and \( \omega \) converge.
### Table 1. Comparison of different methods on classification accuracy (top 1). We use \textit{conv} and \textit{fc} as backbones in the experiments. \dag denotes that the methods have reduced learnable parameters (See Appendix B.1).

<table>
<thead>
<tr>
<th>Model</th>
<th>Tiny ImageNet</th>
<th>STL-10</th>
<th>CIFAR10</th>
<th>CIFAR100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>\textit{conv}</td>
<td>\textit{fc}</td>
<td>\textit{conv}</td>
<td>\textit{fc}</td>
</tr>
<tr>
<td>Fully supervised</td>
<td>36.60</td>
<td>68.70</td>
<td>75.39</td>
<td>42.27</td>
</tr>
<tr>
<td>BiGAN</td>
<td>24.38</td>
<td>20.21</td>
<td>71.53</td>
<td>67.18</td>
</tr>
<tr>
<td>NAT</td>
<td>13.70</td>
<td>11.62</td>
<td>64.32</td>
<td>61.43</td>
</tr>
<tr>
<td>DIM</td>
<td>33.54</td>
<td>36.88</td>
<td>72.86</td>
<td>70.85</td>
</tr>
<tr>
<td>SplitBrain\dag</td>
<td>32.95</td>
<td>33.24</td>
<td>71.55</td>
<td>63.05</td>
</tr>
<tr>
<td>SwAV</td>
<td>39.56 ± 0.2</td>
<td>38.87 ± 0.3</td>
<td>70.32 ± 0.4</td>
<td>71.40 ± 0.3</td>
</tr>
<tr>
<td>SimCLR</td>
<td>36.24 ± 0.2</td>
<td>39.83 ± 0.1</td>
<td>75.57 ± 0.3</td>
<td>77.15 ± 0.3</td>
</tr>
<tr>
<td>CMC\dag</td>
<td>35.41 ± 0.4</td>
<td>40.11 ± 0.2</td>
<td>83.03</td>
<td>85.06</td>
</tr>
<tr>
<td>MoCo</td>
<td>39.90 ± 0.2</td>
<td>41.37 ± 0.2</td>
<td>77.50 ± 0.2</td>
<td>79.73 ± 0.3</td>
</tr>
<tr>
<td>BYOL</td>
<td>41.59 ± 0.2</td>
<td>41.90 ± 0.1</td>
<td>81.73 ± 0.3</td>
<td>81.57 ± 0.2</td>
</tr>
<tr>
<td>Barlow Twins</td>
<td>39.81 ± 0.3</td>
<td>40.34 ± 0.2</td>
<td>80.97 ± 0.3</td>
<td>81.43 ± 0.3</td>
</tr>
<tr>
<td>DACL</td>
<td>40.61 ± 0.2</td>
<td>41.26 ± 0.1</td>
<td>80.34 ± 0.2</td>
<td>80.01 ± 0.1</td>
</tr>
<tr>
<td>LooC</td>
<td>42.04 ± 0.1</td>
<td>41.93 ± 0.2</td>
<td>81.92 ± 0.2</td>
<td>82.60 ± 0.2</td>
</tr>
<tr>
<td>SimCLR + Debiased</td>
<td>38.79 ± 0.2</td>
<td>40.26 ± 0.2</td>
<td>77.09 ± 0.3</td>
<td>78.39 ± 0.2</td>
</tr>
<tr>
<td>SimCLR + Hard</td>
<td>40.05 ± 0.3</td>
<td>41.23 ± 0.2</td>
<td>79.86 ± 0.2</td>
<td>80.20 ± 0.2</td>
</tr>
<tr>
<td>CMC\dag + Debiased</td>
<td>41.64 ± 0.2</td>
<td>41.36 ± 0.1</td>
<td>83.79 ± 0.3</td>
<td>84.20 ± 0.2</td>
</tr>
<tr>
<td>CMC\dag + Hard</td>
<td>42.89 ± 0.2</td>
<td>42.01 ± 0.2</td>
<td>83.16 ± 0.3</td>
<td>85.15 ± 0.2</td>
</tr>
<tr>
<td>MetAug (only OUCL)\dag</td>
<td>42.02 ± 0.1</td>
<td>42.14 ± 0.2</td>
<td>84.09 ± 0.2</td>
<td>84.72 ± 0.3</td>
</tr>
<tr>
<td>MetAug\dag</td>
<td>44.51 ± 0.2</td>
<td>45.36 ± 0.2</td>
<td>85.41 ± 0.3</td>
<td>85.62 ± 0.2</td>
</tr>
</tbody>
</table>

3.4. Model objective

Concretely, we adopt margin-injected meta feature augmentation in the contrastive learning paradigm to achieve desired discriminative multi-view representations, and the proposed $\mathcal{L}_{\text{MetAug}}$ is incorporated to replace the conventional contrastive loss $\mathcal{L}$. The final model objective is defined as:

$$\mathcal{L}_{\text{MetAug}} = \mathcal{L}_{\text{OUC}L}^{\text{ori}} + \delta \cdot \mathcal{L}_{\text{OUC}L}^{\text{aug}}$$ (11)

where $\mathcal{L}_{\text{OUC}L}^{\text{ori}}$ represents the loss \textit{NOT} including the meta augmented features, $\mathcal{L}_{\text{OUC}L}^{\text{aug}}$ represents the loss including such features, and $\delta$ is a coefficient that controls the balance between them (we perform parameter comparisons in Appendix A.4). It is worthy to note that the margin-injected regularization $\mathcal{R}_\sigma$ is only used in meta training the MAGs, i.e., $a_w(\cdot)$, while in regular training of encoders and projection heads, $\mathcal{R}_\sigma$ is discarded. $\mathcal{R}_\sigma$ restricts the augmented features to be informative so that such features can lead the encoder to efficiently and effectively learn discriminative representations. The training process is detailed by Algorithm 1.

4. Experiments

We benchmark our MetAug on five established datasets: Tiny ImageNet (Krizhevsky et al., 2009), STL-10 (Coates et al., 2011), CIFAR10 (Krizhevsky et al., 2009), CIFAR100 (Krizhevsky et al., 2009), and ImageNet (Jia et al., 2009). The compared benchmark methods include: BiGAN (Donahue et al., 2016), NAT (Bojanowski & Joulin, 2017), DIM (Hjelm et al., 2018), SplitBrain (Zhang et al., 2017), CPC
MetAug: Contrastive Learning via Meta Feature Augmentation

Table 2. Performance (accuracy) on the CIFAR10 and STL-10 datasets with ResNet-50 (He et al., 2016).

<table>
<thead>
<tr>
<th>Model</th>
<th>CIFAR10</th>
<th>STL-10</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>SwAV</td>
<td>83.15</td>
<td>82.93</td>
<td>83.04</td>
</tr>
<tr>
<td>SimCLR</td>
<td>84.63</td>
<td>83.75</td>
<td>84.19</td>
</tr>
<tr>
<td>CMC</td>
<td>86.10</td>
<td>86.83</td>
<td>86.47</td>
</tr>
<tr>
<td>BYOL</td>
<td>87.14</td>
<td>87.56</td>
<td>87.35</td>
</tr>
<tr>
<td>Barlow Twins</td>
<td>85.84</td>
<td>86.02</td>
<td>85.93</td>
</tr>
<tr>
<td>DACL</td>
<td>86.93</td>
<td>88.11</td>
<td>87.52</td>
</tr>
<tr>
<td>LooC</td>
<td>87.80</td>
<td>88.62</td>
<td>88.21</td>
</tr>
<tr>
<td>SwAV + Hard</td>
<td>83.99</td>
<td>84.51</td>
<td>84.25</td>
</tr>
<tr>
<td>SimCLR + Hard</td>
<td>86.91</td>
<td>85.48</td>
<td>86.20</td>
</tr>
<tr>
<td>CMC + Hard</td>
<td>88.25</td>
<td>87.79</td>
<td>88.02</td>
</tr>
<tr>
<td><strong>MetAug (only OUCl)</strong></td>
<td><strong>88.79</strong></td>
<td><strong>88.31</strong></td>
<td><strong>88.55</strong></td>
</tr>
<tr>
<td><strong>MetAug</strong></td>
<td><strong>91.09</strong></td>
<td><strong>90.26</strong></td>
<td><strong>90.68</strong></td>
</tr>
</tbody>
</table>

(Hénaff et al., 2019), SwAV (Caron et al., 2020), SimCLR (Chen et al., 2020), CMC (Tian et al., 2019), MoCo (He et al., 2020), SimSiam (Chen & He, 2020), InfoMin Aug. (Tian et al., 2020), BYOL (Grill et al., 2020), Barlow Twins (Zbontar et al., 2021), DACL (Verma et al., 2021) LooC (Xiao et al., 2021), Debaised (Chuang et al., 2020), Hard (Robinson et al., 2020), and NNCLR (Dwibedi et al., 2021).

4.1. Efficiently performing MetAug

**Implementations.** To efficiently perform CL within a restricted amount of the inputs in training, we uniformly set the batch size as 64 (see Appendix A.1 for the comparisons under different settings of batch size). For the experiments with conv and fc as the backbone networks, we adopt a network with the 5 convolutional layers in AlexNet (Krizhevsky et al., 2012) as conv and a network with further 2 fully connected layers as fc. Inspired by the backbone splitting setting of SplitBrain (Zhang et al., 2017), we evenly split the AlexNet into sub-networks across the channel dimension, and each sub-network is the view-specific encoder. The experiments with ResNet-50, we directly change the encoder network to ResNet-50. All backbone encoders are not pre-trained. MetAug (only OUCl) is the ablation variant without margin-injected meta feature augmentation.

Given an RGB image, we convert it to the Lab image color space and split it into L and ab channels. During contrastive learning, RGB, L, and ab are used as three views of the image. Before feeding the views into our model, we simply adopt the same data augmentations in CMC (Tian et al., 2019). Especially, the major contribution of DACL is the proposed data augmentation (i.e., mixup-noise) so that we particularly add mixup data augmentation for DACL. In training, a memory bank (Wu et al., 2018) is adopted to facilitate calculations. We retrieve 4096 past features from the memory bank to derive negatives. The learning rates and weight decay rates are uniform over comparisons.

**Comparison on downstream tasks.** We collect the results of 20 trials for comparisons. The average result of the last 20 epochs is used as the final result of each trial, and the 95% confidence intervals are also reported, while the results without 95% confidence intervals are quoted from the published papers. We compare MetAug against a fully-supervised method (similar to AlexNet (Krizhevsky et al., 2012)) and the state-of-the-art unsupervised methods. Table 1 shows the comparisons on four benchmark datasets. The last two rows of tables represent the results of our methods. As demonstrated in tables, MetAug beats the best prior methods on all datasets. Even compared with the fully-supervised method trained end-to-end (without fine-tuning) for the architecture presented, the proposed method has a significant improvement on most downstream tasks, which demonstrates that MetAug can better model discriminative information when supervision is insufficient (e.g., the training data is limited). The ablation model (i.e., MetAug (only OUCl)) outperforms most unsupervised methods but falls short of the performance of MetAug. Thus, the ablation study proves the effectiveness of our proposed margin-injected meta feature augmentation and optimization-driven unified contrast.

DACL and LooC propose to enhance contrastive learning from the perspective of data augmentation, while MetAug...
improves contrastive learning from the perspective of feature augmentation. The idea behind our method is simple but effective, since contrastive learning works directly on features, and the augmented images need one step of encoding to become features. The experimental results support that MetAug achieves better performance on benchmarks.

Performing MetAug on ResNet. We perform classification comparisons on the CIFAR10 and STL-10 by using ResNet-50. Table 2 shows that MetAug and the ablation variant outperform the compared methods, which indicates that MetAug has strong adaptability to different encoders.

4.2. Benchmarking MetAug on ImageNet

Implementation. To comprehensively understand the performance of our proposed MetAug, we conduct comparisons on ImageNet and make fair comparisons with benchmark methods. The backbone encoder is conv or ResNet-50, and the results are demonstrated in Table 3. MetAug is a decoupled approach so that we can introduce MetAug in the learning paradigm of state-of-the-art to improve the performance, e.g., for the experiments using conv or ResNet-50, we perform MetAug in CMC or NNCLR, respectively.

Results. As shown in Table 3, we find that MetAug can effectively promote the performance of benchmark methods in the comparisons using both conv and ResNet-50. The results support that our proposed meta feature augmentation can enable different encoders to model discriminative information even in the large-scale dataset.

4.3. Is MetAug robust for data augmentation?

To illustrate the impacts of different data augmentations, we conducted multiple comparisons on CIFAR10 shown in Table 4. Note that horizontal flip and rotate are similar, and we use them together in the 1-th comparison. In the 5-th comparison, we take the same data augmentations as the setting of comparisons in Section 4.1. The data augmentations adopted in the 6-th comparison are as same as the setting of

<table>
<thead>
<tr>
<th>ID</th>
<th>horizontal flip</th>
<th>rotate</th>
<th>random crop</th>
<th>random grey</th>
<th>color jitter</th>
<th>mixup</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DACL</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>LooC</td>
</tr>
<tr>
<td>3</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MetAug</td>
</tr>
<tr>
<td>4</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MetAug</td>
</tr>
<tr>
<td>5</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>MetAug</td>
</tr>
<tr>
<td>6</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>MetAug</td>
</tr>
<tr>
<td>7</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>MetAug</td>
</tr>
<tr>
<td>8</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>MetAug</td>
</tr>
</tbody>
</table>

MetAug (only OUCL) w/ Γ
MetAug (only OUCL) w/o Γ

Figure 3. The impact of different \( \phi_{dec} \) on the performance of our method using Γ. Comparisons are conducted on Tiny ImageNet with conv as the encoders.

LooC (Xiao et al., 2021). Additionally, mixup is proposed by DACL (Verma et al., 2021).

We observe from Table 4 that MetAug outperforms the compared methods in all comparisons. It is worth noting that even using weak data augmentation degenerates the performance of our method as well as benchmark methods, but the performance degeneration of our method is minimal compared to others, e.g., from 8-th and 1-th comparison, we find that the gap of MetAug is 1.60%, while that of LooC is 2.44%. The results support that MetAug is robust for various data augmentations.

4.4. Do the variant of \( \Gamma \) promote MetAug?

In practice, we find that the introduction of the weighting factor \( \Gamma \) cannot directly improve our proposed method. Our conjecture lies in that \( \Gamma \) may cause the loss to converge excessively fast, which leaves the network parameters at a local minimum. Therefore, we propose a variant to replace \( \Gamma \) in Equation 9, i.e., \( \Gamma = \frac{\Gamma}{\phi_{dec}} \) where \( \phi_{dec} \) is a linear attenuation coefficient to linearly attenuate the impact of \( \Gamma \) so that the difference between the current value and the optimum becomes smaller.

We use MetAug (only OUCL) to demonstrate the effectiveness of the proposed variant. The results are shown in Figure 3. We observe that the performance of our method get peak value when \( \phi_{dec} = 6 \), which manifests that introducing a certain linear attenuation to \( \Gamma \) can promote MetAug.
5. Conclusion

We conclude that exploring informative features is the key to contrastive learning. Different from the conventional contrastive methods that collect enough informative features to learn a good representation by enlarging the batch or memory bank, we motivate MetAug to learn a discriminative representation from a restricted amount of images. Our method proposes margin-injected meta feature augmentation to straightforwardly augment features to be informative and avoid learning degenerate features. To efficiently make use of all available features, MetAug further proposes optimization-driven unified contrast. Experimental evaluations demonstrate that MetAug achieves the state-of-the-art.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments. This work is supported in part by National Natural Science Foundation of China No. 61976206 and No. 61832017, Key Special Project for Introduced Talents Team of Southern Marine Science and Engineering Guangdong Laboratory (Guangzhou) No. GML2019ZD0603, National Key Research and Development Program of China No. 2019YFB1405100, Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, Beijing Academy of Artificial Intelligence (BAAI), the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China 21XNLG05, and Public Computing Cloud, Renmin University of China. This work is also supported in part by Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the “Double-First Class” Initiative, Renmin University of China, and Public Policy and Decision-making Research Lab of Renmin University of China.

References


A. Appendix - Extended comparisons

In this section, we provide several experimental analyses about the advantages of our proposed method. The experiments to find appropriate hyperparameters are conducted as well, and in detail, we conduct comparisons of using different hyperparameters on the validation set of corresponding benchmark datasets.

A.1. Can MetAug perform consistently under different settings of batch size?

As the results shown in Table 1, 2, and 3, we observe that MetAug achieves our expectation that learning anti-collapse and discriminative representations from a restricted amount of images in a training step (i.e., the batch size is limited). However, we conduct further experiments to explore whether MetAug has consistent performance under settings of larger batch sizes.

From Figure 4, we observe that with the increase of batch size, each compared method achieves better performance on the downstream task. We conjecture that with the enlarging of batch size, the number of available features in a training step is increased, so that models may explore more informative features to promote the performance of contrastive learning. Yet, comparing our method with the benchmark methods, we find that LooC leverages more than one (e.g., three) contrastive loss in a training step, which allows LooC to train the model multiple times. We observe that, even in the large batch size, MetAug can still improve the state-of-the-art methods by a significant margin.

Concretely, MetAug maintains its superiority over compared method under different settings of batch size.

A.2. Variants of the injected margin

We denote \( \min^+ \) as \( \min(\{d(\{z^+\}_{k})\}) \) and \( \max^- \) as \( \max(\{d(\{z^-\}_{k})\}) \). For "Medium", both \( \sigma^+ \) and \( \sigma^- \) equal to \( \text{mean}[\min^+, \max^-] \). \( \sigma^+ = \max[\min^+, \max^-] \) and \( \sigma^- = \min[\min^+, \max^-] \) in "Small".

In Figure 5, we conduct comparisons on CIFAR100 with fc. We observe that, whether our method uses \( \mathcal{L}_{\text{contrast}} \) or the proposed \( \mathcal{L}_{\text{OUCL}} \), all three variants can improve MetAug, and our method with "Large" achieve the best performance. The experiments further prove the effectiveness of the two key ingredients of MetAug.

A.3. Understanding of the augmented features

To understand the augmented features, we conduct a comparison of MetAug by adopting the augmented features in
We conducted comparisons based on MetAug (only OUCL) on Tiny ImageNet with fc encoder. To measure the impacts, we iteratively fixed one parameter and then study on the others by selecting them in the ranges, respectively.

As shown in Table 5, the results of MetAug using the augmented features in the test are listed in the w/ augmented features column, and the results of MetAug NOT using the augmented features in the test are listed in the w/o augmented features column (which is the regular approach in the test). We select δ from the range of \(\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}\) to generate different augmented features. Specifically, the approach of adopting the augmented features in the test is that we use MAGs to generate augmented features and such features are treated as the same as the original features, i.e., augmented features are regarded as additions of original features. Note that, in regular test (i.e., w/ augmented features), we use the representation \(h_i^\beta\) to feed into the classifier, while, in the test of using augmented features, we have to use the feature \(z_i^\beta\) generated by the projection head \(g_{\beta}(\cdot)\) to feed into the classifier, because MAGs work on the feature \(z_i^\beta\).

From Table 5, we observe that, generally, MetAug w/o augmented features beats MetAug w/ augmented features. The reasons behind such phenomenon are: 1) the augmented features are generated to lead the encoders to learn discriminative representations (e.g., \(h_i^\beta\)), which indicates that the augmented features contribute to the improvement of the encoders, but this does not mean that the augmented features are discriminative for downstream tasks; 2) in the test of using augmented features, we do not discard the projection head \(g_{\beta}(\cdot)\), and recent works prove that the approach of using a projection head in training and discarding such head in the test can significantly improve the performance of the model on downstream tasks (Chen et al., 2020; He et al., 2020).

Out of the understanding of the experimental results, we think that the augmented features contain useful information that can improve the encoder, but such information may not be discriminative to downstream tasks.

### A.4. Synthetic comparison of hyperparameters

To intuitively understand the impacts of hyperparameters, we conduct comparisons by using various combinations of them for the proposed MetAug. Specifically, \(\alpha\) controls the impact of the proposed margin-injected regularization term. The hyperparameter \(\beta\) is proposed as a temperature coefficient in OUCL. \(\gamma\) is a specific parameter to replace the hyperparameters in OUCL such that the number of hyperparameters can be reduced. \(\delta\) balances the impact of OUCL that uses augmented features and OUCL that does not use these features.

As demonstrated in Figure 7, we first solely study \(\gamma\)'s impact on MetAug, because \(\gamma\) is only used in OUCL function, and in practice, we find that, compared with other hyperparameters, \(\gamma\) has less impact on our method. We conduct experiments on Tiny ImageNet with fc encoder and select \(\gamma\) from the corresponding range for MetAug (only OUCL) to clarify its impact, and the results indicate that appropriate selected \(\gamma\) can indeed promote the performance of our
method, but the differences between the impacts of different γ are limited.

Then, we fix γ = 0.40 and study on the impacts of other hyperparameters. As the results are shown in Figure 6, the plots further elaborate our parameter studies’ results with MetAug on the CIFAR10 benchmark dataset with fc encoder. To explore the influence of β and δ, we first fixed α = 10^{-13}, and then we selected β from the range of \{2, 2^2, 2^4, 2^5, 2^6, 2^7, 2^8\} and δ from the range of \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}. Following the same experimental principle as above, we selected α from the range of \{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}, 10^{-11}, 10^{-12}, 10^{-13}, 10^{-14}, 10^{-15}, 10^{-16}\}.

See Figure 6(a), (b), and (c) for the details of the comparisons. In general, good classification performance highly depends on the β and δ terms. Also, α is an intensely necessary supplement for adapting the interval between similarities of augmented features and original features, which avoids to learn degenerate representations. We also find that the potential to improve the learned representations grows with the adjustment of term β, e.g., the initial loss becomes relatively large.

### A.5. Discussion of the comparisons using ResNet-50 on ImageNet

In Table 3, we do not report the experimental results of CMC using ResNet-50, because the batch sizes adopted by them are inconsistent, i.e., CMC adopts 128 yet other methods adopt 4096. Therefore, the performance of CMC is not competitive compared to other compared methods, including MetAug\(^\ast\). Our proposed MetAug can be treated as a feature augmentation approach so that MetAug can be embedded into each self-supervised learning architecture. The original MetAug is implemented based on CMC, so for a fair comparison, MetAug adopts the same batch size as CMC, while MetAug\(^\ast\) is implemented based on NNCLR (Dwibedi et al., 2021) with the batch size of 4096. Therefore, MetAug\(^\ast\) apparently outperforms MetAug, so we only report the best variant of our method, i.e., MetAug\(^\ast\)\. Whereas, we have comprehensively evaluated whether our method can really improve the performance of CMC on ImageNet, which is shown in Table 6. MetAug can be treated as a component to improve baseline methods, which is implemented based on CMC\(^\dagger\). We observe that MetAug beats CMC and even achieves better performance than CMC + RandAugment, which proves that our method can not only improve CMC but also beat RandAugment. We further employ RandAugment in our method. The results show that such a variant achieves the best performance, which is consistent with 4.3, i.e., stronger data augmentation improves the performance of our method.

### B. Appendix - Implementation

In this paper, we introduce a novel self-supervised representation learning approach, i.e., Meta Feature Augmentation (MetAug), of which Figure 1 depicts the overview framework. The following subsections provide the design details of MetAug.

#### B.1. Network architecture

In the experiments, neural network classification methods (i.e., conv and fc) are adopted as the backbone networks, and the classifiers (i.e., the linear networks) on the representations extracted from the encoders are performed on downstream classification tasks.

According to the principle of building the encoders, the AlexNet is split across the channel dimension with the idea that split-AlexNet can also perform well in learning representations between views, which only has the halved learnable parameters (Zhang et al., 2017). We build the AlexNet with 5 convolutional layers, 2 linear layers, and a fully connected layer followed by a l2 normalization function. Then the split-AlexNets (i.e., the sub-networks) are regarded as the encoders. In experiments, we use conv and fc, which use the corresponding layers of AlexNet. Note that we split AlexNet across channels for RGB, L, and ab views. in the test, we concatenate representations layer-wise from the encoders into one to achieve the final representations of the inputs.

We develop the classifier by leveraging a linear network followed by a softmax output function. Following the proposed experimental setting of the previous literature (Oord et al., 2018; Hjelm et al., 2018; Arora et al., 2019; Tian et al., 2019), we evaluate the quality of the learned representations by freezing the weights of backbone encoders and training the linear classifier in the test.
B.2. Algorithm description

MetAug is an end-to-end representation learning method: we iteratively train the encoders and MAGs by back-propagating $L_{\text{MetAug}}$, and the training process is based on Adam gradient optimization.

The proposed MetAug is a generalized approach, which can be used for various downstream tasks, e.g., classification, clustering, regression, etc. We can straightforwardly train the encoders, pretrained by MetAug, on downstream tasks.

Here, we provide a pseudo-code for MetAug training described in the style of PyTorch, which is without the inclusion of the detailed matrix processing or helper utility functions & codes that are irrelevant to the algorithm: