MIM-CLR: Contrastive Representation Learning for Masked Image Modeling

Roberto Halpin-Gregorio, Dhruv Sreenivas, Simar Kohli, Eliot Shekhtman
Department of Computer Science, Cornell University
Ithaca, NY 14853
{rgh224, ds844, sk2523, ess239}@cornell.edu

Abstract

Recently, masked image modeling has become a hot area of research in computer vision. The main inspiration for this class of approaches is the success of masked word modeling in NLP, where models such as BERT [6] and the GPT line [2] have utilized this method for state-of-the-art results in language modeling. However, recent methods have seen limited success on image classification tasks, including SimMIM [20], which achieves only a 56.7% linear top 1% evaluation accuracy on ImageNet-1K. We propose to learn better frozen image features by introducing MIM-CLR, a model that minimizes a novel joint loss between the SimMIM reconstruction loss and a scaled contrastive loss. The contrastive loss aims to push features of masked images and their corresponding ground truth images close together, while pulling apart the features of every other pair learning better representations. Code for our model and experiments can be found at https://github.com/roberto-hg/Contrastive-SimMIM.

1 Introduction

Image representation learning is a critical field within computer vision research, and methods to learn meaningful and compact representations have been explored in tandem with other representation research such as language modeling.

The field has recently been dominated by contrastive approaches. These approaches augment images and seek to learn representations such that the representations of a single image under different augmentations are close while representations of an augmented image to different augmented images are further apart. This is meant to learn invariance to a variety of augmentations, and in so learn meaningful differences between images for downstream fine supervised approaches. This often comes with the issue of humans needing to decide which augmentations are more or less meaningful for downstream tasks than others. Standard augmentations applied in state of the art model SimCLR [3] involve random cropping, color distortion, and Gaussian blur.

Masked image modeling is a generalizable image representation learning scheme inspired by the success of large-scale masked language models in natural language processing. The hope is that representations learned through masked image modeling can lead to few-shot generalization across a wide variety of image-based downstream tasks, similar to how large MLMs like BERT [6] and GPT-3 [2] are universally used throughout the area of natural language processing. As Richard Feynman once said, "what I cannot create, I do not understand," which implies that recreating images from their masked counterparts can potentially lead to better general understanding of them, which can be used in downstream tasks.

Generally, masked image modeling applies a mask to patches of an image, and during training, a linear head is applied to the model feature representations to predict raw pixel values for pixels within...
each patch, and a loss is applied with true pixel values as the self-supervised labels. Representations learned by models such as SimMIM [20] have exhibited strong results on a variety of downstream tasks with fine-tuning, competitive with their mainstream, contrastive counterparts. Unfortunately, SimMIM representations without fine-tuning failed to perform comparably: whereas fine-tuning marginally outperformed other methods, linear probing yielded drastically worse results. On the other hand, transformer-based contrastive methods, such as MoCo v3 [4], have achieved significantly higher accuracies.

We seek to combine both these methods to extend the SimMIM architecture, a reconstruction-based masked image modeling approach, by incorporating a contrastive loss when learning features along with the SimMIM loss. It is hypothesized that jointly minimizing this contrastive loss along with the SimMIM reconstruction loss will result in features that are invariant to how we mask the images, allowing the model to generalize and perform better on unseen data while retaining the deep semantic feature learning benefits from masked image modeling.

2 Related Work

2.1 Self-Supervised Representation Learning

Unfortunately, traditional methods of self-supervised learning have fallen behind the comparable performances seen in other forms of machine learning and models. This has been especially true within areas of increasing dataset sizes [12]. However, the introduction of contrastive learning approaches within traditional modeling pipelines have pushed the envelope of performance [19]. While some proposals have been seen for utilizing alternatives to contrastive objectives, the application of contrastive learning within self-supervised learning still has much to offer.

2.2 Contrastive Representation Learning

A lot of the recent self-supervised learning literature across a variety of fields has focused on contrastive learning methods, which essentially boil down to the following learning regime: given positive pairs of inputs (i.e. transformations of the same image/state in computer vision/RL) and negative pairs (i.e. transformations of different inputs), we want to learn a feature map \( f_\theta \) that makes sure positive pairs are close together in feature space, while negative pairs are far apart in said space, thereby “contrasting” features of different inputs.

The CPC line [14] of methods introduced the InfoNCE contrastive loss, and showed that learning CPC representations across a variety of modalities, especially in computer vision, improved the top-1 and top-5 accuracies of linear classifiers trained on top of said learned representations. SimCLR [3] built on top of the InfoNCE loss by transforming input images \( x \) into two new views \( x', x'' \) (done via simple augmentations such as random cropping), as opposed to the patch autoregressive method in [14]. This method of augmenting the input image gives us a natural definition of positive and negative examples. SimCLR’s similarity function is a standard cosine similarity loss.

2.3 Masked Language Modeling

Masked language modeling has been the dominant method for unsupervised representation learning in natural language processing over the past few years. It builds off of the next token prediction task [15], which takes in a sequence of tokens and seeks to predict the next token in sequence, and the continuous bag of words task [11], which takes in a window of tokens and seeks to predict the middle token. The unifying feature of these tasks is that they all model relationships between tokens and proximity, claiming that by knowing the relationships between tokens and their contexts we can capture useful structural and semantic information in language that could be leveraged for downstream tasks.

Masked language modeling takes in a sequence of tokens and masks out tokens randomly for prediction, learning the token’s relationship to its bidirectional context. Since seeing great success 4 years ago with the BERT model [6] in pre-training large language models and exhibiting strong transfer learning performance to a variety of tasks, it has inspired similar methods in a variety of fields.
2.4 Masked Image Modeling

Masked image modeling builds off of the core concept of masked language modelling, seeking to learn meaningful image representations by predicting on masked patches of an image given the rest of the image, learning long-distance dependencies between pixels depending on patch sizes. Given that language has natural discrete units in tokens, many masked image modelling tasks sought to similarly use meaningfully segmented image patches. An example of this can be seen in the BEiT [1] model, which uses a combination of 16x16 patches and discrete tokens obtained through a dVAE image tokenizer. SimMIM [20] presented a simpler framework, removing tokenization but achieving competitive results. We will be using this model as a base for masked image modelling.

2.5 Joint loss minimization

Multiple forays have been made into the application of more advanced representation-learning models, a variety of which are trained via the minimization of a combination of losses, which can be either task-specific or data-specific. ALIGN, one of the foundational works in representation learning aimed to utilize noisy image alt-text data to generate visual/language representations [9]. The MURAL team decided to extend the ALIGN model to the multi-lingual setting and modified the model by adding a cross-lingual objective [8]. This improved results on image and text retrieval.

SLIP [13] applies two separate methods (CLIP [16] and SimCLR) in order to merge image self-supervision and language supervision. While the team notes that it is not immediately clear why these would generate a stronger performance [13]. We similarly merge SimMIM and SimCLR, in attempt to see whether performance can be similarly increased or not.

3 MIM-CLR Framework

Our approach to masked image modeling was to minimize a novel joint loss utilizing a contrastive loss to get more disentangled features for downstream classification tasks. Given a masked image \( x \), we apply a "weak" augmentation (i.e. random cropping or shifting) and a "strong" augmentation to the image (i.e. the weak augmentation + the masking). The main idea is to make the representations of the weak and strong image augmentations similar, while representations for different images should be farther apart. We do this by applying the SimCLR [3] InfoNCE loss from SLIP [13] to the weakly augmented representations \( z \) and the masked augmented representations \( z_{\text{mask}} \). In total, MIM-CLR’s loss function is a linear combination of both the reconstruction SimMIM loss and a contrastive InfoNCE loss:

\[
\mathcal{L}_{\text{MIMCLR}} = \mathcal{L}_{\text{reconstruction}} + \lambda \cdot \mathcal{L}_{\text{contrastive}}.
\]
In some ways, our approach can be seen as a self-supervised version of FixMatch [18], which learns an image classification model in the semi-supervised learning setting (some labels, but not all) by "matching labels" of weakly augmented and strongly augmented versions of the same images together. However, there are a couple of key differences:

1. FixMatch operates in the semi-supervised learning setting, so it incorporates a standard cross entropy loss on the labeled data that we cannot use in our setting.
2. FixMatch assigns "pseudo-labels" to unlabeled images if the model’s confidence of the classes on those images is above a certain threshold \( \tau \). This pseudo-label is then used to help match the predicted labels on weakly and strongly augmented versions of the same unlabeled data. However, this is different from our setting, as the validity of these pseudo-labels is due in part to having labels to train on anyway, which we do not have.

3.1 SimMIM Method

SimMIM [20] is a recent masked image modeling method that focuses on reconstructing the images in question by minimizing an \( \ell_1 \) loss between the real image input \( x \) and the predicted image output \( y \). One can see this \( \ell_1 \) loss as a measure of how close the RGB pixel values of the masked pixels were to the corresponding real image pixels.

To do this reconstruction, the authors used an autoencoder-based architecture, where the encoder was either a Swin transformer [10] or a vision transformer [7]. The transformer architecture, due to its success in masked language modeling in NLP, was used in this case. The decoders were an inverse Swin transformer and an inverse vision transformer, respectively. See the left side of Figure 1 for an illustration of this process. The authors found that the \( \ell_1 \) loss was the most suitable to achieve state-of-the-art results in a variety of computer vision tasks (image classification, object detection, semantic segmentation) with significantly less labeled data.

3.2 SimCLR Method

SimCLR [3] is a well-known, simple contrastive method to learn useful representations for image classification. The main idea behind SimCLR is that different augmentations of one image \( I \) (positive samples), should be "similar" in the feature space, whereas different images (regardless of augmentation) should be separate (negative samples). These augmented images first get passed through a feature encoder, and then a MLP projector (see the right side of Figure 1). SimCLR has been shown to outperform a variety of other image classification models on ImageNet ILSVRC-2012 [17] with significantly less parameters than other competitive models.

3.3 MIM-CLR (our method)

Our method (shown in Figure 1), Masked Image Modeling through Contrastive Learned Representations (MIM-CLR) seeks to combine the strengths of masked image models like SimMIM and contrastive classification models like SimCLR to improve linear evaluation image classification results with SimMIM.

As mentioned previously, SimMIM notably performs worse in comparison to other models in terms of linear evaluation top 1% accuracy on ImageNet-1K, which we hypothesize is the result of learning representations invariant to just how we can mask the image. We believe we can improve upon these features by adding a contrastive loss to the SimMIM objective.

4 Implementation

We made small changes to the original SimMIM repository, adding in our contrastive loss to the SimMIM module and support for the STL-10 dataset. Psuedocode for our MIM-CLR model is found in Algorithm 1.

Due to the scope of this work, we did not train on ImageNet, but instead on the STL-10 dataset [5]. This dataset is inspired by CIFAR-10, but is more conducive to unsupervised representation learning, as there is a significant amount of unlabeled data in the dataset. STL-10 has 100k unlabeled images, 5k labeled training images, and 8k testing images all of size 96x96x3. When pre-training we use
Algorithm 1 MIM-CLR: PyTorch-like pseudocode

```python
# encoder: vision transformer encoder network
# lambda_: Contrastive (SimCLR) scaling hyperparameter
def forward(img, mask):
x, x_mask = img, apply_mask(img, mask)
z, z_mask = encoder(x), encoder(x_mask) # vision transformer embed: N x C
loss = lambda_ * simclr(z, z_mask) + simmim(x, z_mask)
return loss

# decoder: vision transformer decoder network
def simmim(x, z_mask, mask):
x_rec = decoder(z_mask) # Reconstruct image
# Reshape mask to be compatible with loss
mask = mask.repeat_interleave(patch_size, 1).repeat_interleave(patch_size, 2).unsqueeze(1).contiguous()
loss_recon = l1_loss(x, x_rec)
loss = (loss_recon * mask).sum() / (mask.sum() + 1e-5) / color_channels # usually 3
return loss

# tau: softmax temperature
def simclr(z1, z2):
z1, z2 = normalize(z1, z2)
label = range(N)
mask = eye(N) * 1e9
logit = z1 @ z2.T
logit1 = z1 @ z1.T - mask
logit2 = z2 @ z2.T - mask
logit1 = cat(logit, logit1)
logit2 = cat(logit.T, logit2)
l1 = CrossEntropy(logit1 / tau)
l2 = CrossEntropy(logit2 / tau)
loss = (l1 + l2) / 2
return loss
```

Notes: `@` is the matrix multiplication operator. `k.T` is `k`’s transpose. `eye` constructs an identity matrix. `cat` concatenates two matrices. Figure adapted from SLIP [13].

the combination of the unlabeled images and the labeled training images (without labels), giving us 105k unlabeled images. We pre-trained our MIM-CLR models with different contrastive scaling hyperparameters $\lambda$ and also trained a model by minimizing only the contrastive loss and not the masked modeling loss. All our training hyperparameters were copied from standard SimMIM setups. Note that these hyperparameters were primarily used for ImageNet experiments, so using them in both another model setup and a different dataset can lead to unpredictable results. However, due to time constraints we were unable to tune any hyperparameters other than the constrastive scaling ($\lambda$). Check Appendix A.1 for more training details.

There are two main protocols for evaluating visual self-supervised models:

- **Linear Evaluation**: The main idea behind linear evaluation is to completely freeze the pre-trained feature network and attach a linear head on top. This linear head is then trained for classification on the labeled train dataset with Cross Entropy loss. Evaluating this model on the test set, gives us our desired accuracy metric.

- **Fine-tuning**: In fine-tuning we also have a linear heap on top of the pre-trained feature network. However, the key difference is that the feature network is not frozen. This full pipeline is trained on the small labeled train set, usually with a much smaller learning rate, with the same classification framework as linear evaluation.

5 Experimental Results

We compare MIM-CLR’s top 1% and top 5% linear evaluation and fine-tuning scores to SimMIM’s on STL-10. We pre-train all our models for 100 epochs on the 105k training images, saving models every 5 epochs. Using these save models we use both evaluation protocols: fine-tuning and linear
Table 1: Our method, MIM-CLR, outperforms SimMIM in both linear evaluation and fine-tuning protocols on the STL-10 test set. Mean and standard deviation accuracies (top-1 and top-5) for both evaluation protocols.

<table>
<thead>
<tr>
<th></th>
<th>Linear Eval</th>
<th>Linear Eval</th>
<th>Fine-tune</th>
<th>Fine-tune</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top-1 acc (%)</td>
<td>Top-5 acc (%)</td>
<td>Top-1 acc (%)</td>
<td>Top-5 acc (%)</td>
</tr>
<tr>
<td>SimMIM [20]</td>
<td>28.53 ± 0.42</td>
<td>80.30 ± 0.11</td>
<td>63.51 ± 0.16</td>
<td>97.20 ± 0.08</td>
</tr>
<tr>
<td>MIM-CLR (ours)</td>
<td>38.84 ± 0.64</td>
<td>89.04 ± 0.24</td>
<td>65.73 ± 0.68</td>
<td>97.45 ± 0.12</td>
</tr>
</tbody>
</table>

Figure 2: Our best performing method variant ($\lambda = 5.0$) consistently outperforms SimMIM ($\lambda = 0.0$) regardless of the amount of epochs spent pre-training in both fine-tuning and linear evaluation.
Figure 3: Our method outperforms SimMIM ($\lambda = 0.0$) regardless of the amount of epochs spent pre-training in almost all cases. A contrastive scaling ($\lambda$) of 5.0 tends to perform best across the board. However, having no reconstruction loss component causes the model to degrade poorly in linear evaluation, and perform very poorly when fine-tuning.

6 Discussion & Conclusion

This report presents a simple way to increase the performance of masked image models in the case of image classification, through the addition of a simple contrastive loss to the standard masked loss. Our results suggest that MIM-CLR consistently outperforming SimMIM on STL-10 can translate over to a more large-scale, higher-resolution image classification setting, such as in ImageNet. However, there are plenty of future questions and open discussion areas:

- It is clear that training the model on small images (96 x 96 in the STL-10 case) hurts the reconstruction abilities of the model in the future, as minimizing a reconstruction loss doesn’t necessarily distill any useful information into the model (fine details get lost). For example, it is more impressive and difficult to reconstruct a 224 x 224 image rather than a 96 x 96 image. This is our hypothesis for why SimMIM did not improve much regardless of the number of epochs we trained.
- We did not tune any hyperparameters in our experiments, thus opting to use the ImageNet-optimal SimMIM hyperparameters to minimize the reconstruction losses. This has the potential to be improved for the case of STL-10 in particular.
- Compared to more standard contrastive learning objectives, such as SimCLR [3], the data augmentations used in MIM-CLR are completely different from the augmentations generally used in SimCLR. This goes to show why just a pure contrastive objective doesn’t work as well in our experiments (in our case, it doesn’t really get minimized at all, see Figure 4). We believe that a standard SimCLR model, where the same type of augmentation (i.e. different random crops) is performed on positive pairs of images, has the potential to do very well on STL-10 in terms of the accuracies reported in this paper.

Given these discussion points, we believe there is a lot of room for improvement in our method, especially improving upon our current 38.8% linear evaluation top-1% accuracy.

References


A Appendix

A.1 Experiment Details

<table>
<thead>
<tr>
<th>Model</th>
<th>Swin-custom</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Channel</td>
<td>64</td>
</tr>
<tr>
<td>Depths</td>
<td>{2, 2, 18, 2}</td>
</tr>
<tr>
<td>Heads</td>
<td>{4, 8, 16, 32}</td>
</tr>
<tr>
<td>Params</td>
<td>23M</td>
</tr>
<tr>
<td><em>Pre-training</em></td>
<td></td>
</tr>
<tr>
<td>Input Size</td>
<td>96</td>
</tr>
<tr>
<td>Window Size</td>
<td>6</td>
</tr>
<tr>
<td>Mask Patch Size</td>
<td>16</td>
</tr>
<tr>
<td>Mask Ratio</td>
<td>0.6</td>
</tr>
<tr>
<td><em>Fine-tuning</em></td>
<td></td>
</tr>
<tr>
<td>Input Size</td>
<td>96</td>
</tr>
<tr>
<td>Window Size</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 2: Detailed architecture specifications.

<table>
<thead>
<tr>
<th>Train</th>
<th>pre-train</th>
<th>Lin Eval</th>
<th>Fine-tune</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epochs</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Warmup Epochs</td>
<td>10</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Base LR</td>
<td>2e-4</td>
<td>1.25e-3</td>
<td>1.25e-3</td>
</tr>
<tr>
<td>Warmup LR</td>
<td>1e-6</td>
<td>2.5e-7</td>
<td>2.5e-7</td>
</tr>
<tr>
<td>Min LR</td>
<td>1e-5</td>
<td>2.5e-7</td>
<td>2.5e-7</td>
</tr>
<tr>
<td>Weight Decay</td>
<td>0.05</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Layer Decay</td>
<td>0</td>
<td>0.9</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 3: Detailed training specifications.
A.2 Additional Experimental Results

Figure 4: **Unscaled loss** values during pre-training on STL-10 unlabeled+train. In the first subplot: there are slight variations in unscaled contrastive loss, higher contrastive scaling tends to produce smaller contrastive loss. Interestingly, the second subplot shows that reconstruction loss is essentially unaffected by contrastive scaling ($\lambda$). In the bottom two subplots: when including the no reconstruction loss or no contrastive loss variants we observe very poor reconstruction and contrastive loss respectively, which is as expected.
Figure 5: When looking at the joint loss of our models, almost all models perform similarly. However, when reconstruction loss is not considered (subplot two) the model quickly converges to near zero loss. This is most likely due to an easy contrastive objective, as most models get near zero loss (Figure 4).

Figure 6: Similar results are observed when measure top-5 accuracy on STL-10 test set. These are the top-5 accuracy ablation results across pre-training epochs.