PiLaMIM: Toward Richer Visual Representations by Integrating Pixel and Latent Masked Image Modeling

Junmyeong Lee **Eui Jun Hwang** **Sukmin Cho** **Jong C. Park**
School of Computing
Korea Advanced Institute of Science and Technology
{david516, ehwa20, nelllpic, jongpark}@kaist.ac.kr

Abstract

In Masked Image Modeling (MIM), two primary methods exist: Pixel MIM and Latent MIM, each utilizing different reconstruction targets, raw pixels and latent representations, respectively. Pixel MIM tends to capture low-level visual details such as color and texture, while Latent MIM focuses on high-level semantics of an object. However, these distinct strengths of each method can lead to suboptimal performance in tasks that rely on a particular level of visual features. To address this limitation, we propose PiLaMIM, a unified framework that combines Pixel MIM and Latent MIM to integrate their complementary strengths. Our method uses a single encoder along with two distinct decoders: one for predicting pixel values and another for latent representations, ensuring the capture of both high-level and low-level visual features. We further integrate the `[CLS]` token into the reconstruction process to aggregate global context, enabling the model to capture more semantic information. Extensive experiments demonstrate that PiLaMIM outperforms key baselines such as MAE, I-JEPA and BootMAE in most cases, proving its effectiveness in extracting richer visual representations. The code is available at https://github.com/joonmy/PiLaMIM.git.

1 Introduction

Masked Image Modeling (MIM) has shown to be an effective pre-training method for Vision Transformers (ViT) [7], achieving notable successes in various downstream tasks. MIM involves masking a significant portion of an image and training the model to predict the masked regions based solely on the visible context, using various reconstruction targets such as pixels [8, 9, 17, 21], latent representations [1, 2, 5, 24], discrete visual tokens [3, 15], and hand-crafted features [19]. The reconstruction target in pre-training plays a crucial role in determining representation quality, directly aligned with high performance of certain downstream tasks [1, 3, 13, 15, 19, 20].

In particular, when pre-trained with pixels (**Pixel MIM**) and latent representations (**Latent MIM**) as the reconstruction targets, the visual representations learned by the two methods tend to have contrasting characteristics [1, 20]. Pixel MIM focuses on low-level details such as color, edges and texture, making it particularly effective for tasks that require fine-grained visual information such as object counting and depth prediction. By contrast, Latent MIM captures high-level semantics, excelling at tasks such as image classification, where abstract and semantic understanding of an object is more important.

However, those specialized strengths of each MIM method can also be viewed as obvious weaknesses: (i) Pixel MIM, with its emphasis on low-level details, is limited to capture high-level semantics. This results in lower performance on tasks that require deeper semantic understanding, especially without

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

fine-tuning [1, 9, 13, 20]. (ii) Latent MIM, on the other hand, uses latent representations as learning targets, where pixel details are potentially removed. As a result, it tends to perform worse than Pixel MIM in tasks that rely on low-level details [1, 20]. These limitations indicate that neither Pixel MIM nor Latent MIM is entirely optimal on its own, but their strengths may complement each other's weaknesses in understanding the visual context. This presents an opportunity to achieve richer visual representations by combining both reconstruction targets in pre-training ViT [7] with MIM.

Building on these observations, we integrate both **Pi**xel MIM and **La**tent MIM methods into a unified framework, **PiLaMIM**, which complements each method and enables richer visual representations. This structure employs a shared context encoder along with two distinct decoders, each serving a different purpose: one designed to reconstruct pixel values, and the other focused on predicting latent representations. Moreover, we enhance the prediction of latent representations by incorporating the `[CLS]` token in the process, following iBOT [24]. The inclusion of the `[CLS]` token is critical, as it aggregates global context across the entire image, encouraging the model to go beyond local patches and capture more abstract, semantically rich information [15, 24].

To highlight the importance of capturing both high-level and low-level visual features, we design our experiments in two distinct categories: (i) image classification tasks that rely on high-level semantics using ImageNet-1K [16], CIFAR10 [12], CIFAR100 [12], iNaturalist2021 [10], and Places365 [23], and (ii) object counting and depth prediction tasks that depend on low-level details using Clevr/Count [11] and Clevr/Dist [11]. We experimentally show that PiLaMIM consistently outperforms MAE [9] and I-JEPA [1], which are leading methods in Pixel MIM and Latent MIM, respectively. Moreover, PiLaMIM shows better performance than BootMAE [6], which similarly attempts at integrating both MIM methods. These results demonstrate that our method excels at extracting richer visual representation by effectively capturing both high-level and low-level visual features. Furthermore, we find that simply utilizing the `[CLS]` token in the reconstruction process can significantly enhance the performance in both high-level and low-level visual tasks.

2 Method

To capture both high-level and low-level visual features, as illustrated in Figure 1, we extend the Masked Autoencoder (MAE) [9] by integrating two distinct decoders: a pixel decoder and a latent decoder. The pixel decoder is responsible for reconstructing at the pixel level, while the latent decoder focuses on reconstructing latent level features. The specifics of each component in our method are described in the following sections.

Context Encoder. The input image I is initially divided into N non-overlapping patches $X = \{x_i\}_{i=1}^{N}$, where x_i denotes the i-th patch of the image. A portion k of these patches is randomly selected to be masked, resulting in two sets: visible patches $X_{\mathcal{V}} = \{x_i\}_{i \in \mathcal{V}}$ and masked patches $X_{\mathcal{M}} = \{x_i\}_{i \in \mathcal{M}}$, where $\mathcal{V}$ and $\mathcal{M}$ represent the index sets of the visible and masked patches, respectively. The context encoder f_{context} processes only the visible patches, which are first linearly projected to match the encoder's dimensions. Then, positional embeddings $P_{\mathcal{V}}$ are added along with the `[CLS]` token prepended. These processed inputs are subsequently fed into the ViT [7] layers to output the latent representation $Z_{\mathcal{V}}$:

$$Z_{\mathcal{V}} = f_{\text{context}}(\texttt{[CLS]}, X_{\mathcal{V}}, P_{\mathcal{V}}) \tag{1}$$

Target Encoder. The target encoder f_{target} shares the same structure as the context encoder and is initialized with the same weights. It is updated using the exponential moving average (EMA) of the context encoder weights [4], following the update rule: $\theta_{\text{target}}^{(t)} = \lambda \theta_{\text{target}}^{(t-1)} + (1-\lambda)\theta_{\text{context}}^{(t)}$, where λ is the momentum value and θ indicates model parameters. The target encoder f_{target} takes the entire patched image X along with positional embedding P and the `[CLS]` token as input to extract the target latent representation $T = \{t_i\}_{i=0}^{N}$:

$$T = f_{\text{target}}(\texttt{[CLS]}, X, P) \tag{2}$$

Pixel and Latent Decoder. The decoder consists of two components: pixel decoder g_{pixel} for pixel reconstruction and latent decoder g_{latent} for latent representation reconstruction. After the encoding process, the encoded tokens are fed into each decoder and linearly projected to match the decoder's dimensions. These tokens and the learnable tokens $M_{\mathcal{M}}$ and $N_{\mathcal{M}}$, which are introduced to predict the

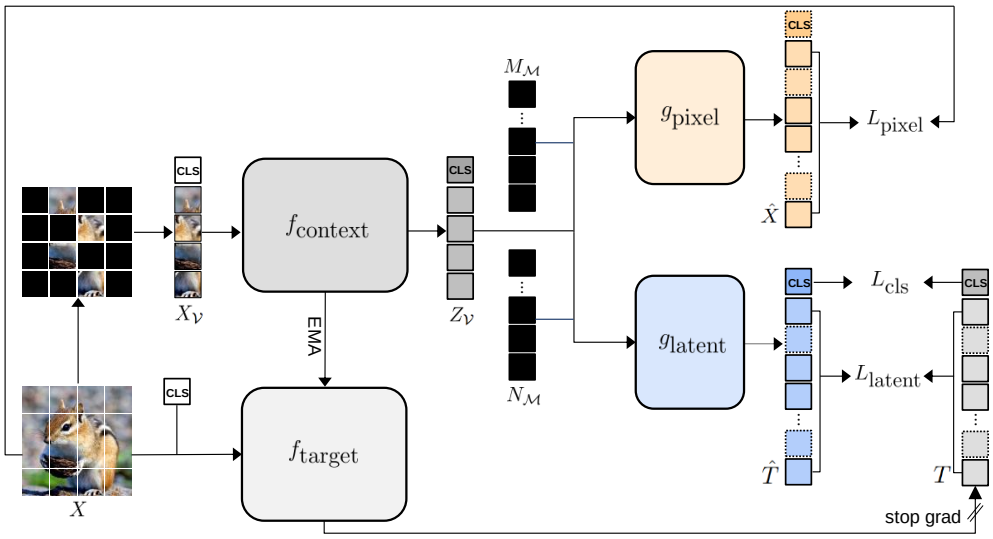


Figure 1: An overview of the PiLaMIM framework. The context encoder f_{context} processes the visible patches $X_{\mathcal{V}}$ to produce encoded tokens $Z_{\mathcal{V}}$. These tokens are then fed into two decoders: the pixel decoder g_{pixel} and g_{latent}, resulting in predicted pixel values $\hat{X}$ and latent representations $\hat{T}$, respectively. The target encoder f_{target} processes the patched image X to provide target latent representation T. For training, we define three loss functions: L_{pixel} and L_{latent} for masked patches, and L_{cls} for the `[CLS]` token.

masked patches $X_{\mathcal{M}}$, are rearranged back to their original position, after which positional embeddings P are added. This is then passed through ViT [7] layers and projected again to match each target's dimensions, generating the predicted pixel values $\hat{X}$ and latent representations $\hat{T}$:

$$\hat{X} = g_{\text{pixel}}(Z_{\mathcal{V}}, M_{\mathcal{M}}, P), \quad \hat{T} = g_{\text{latent}}(Z_{\mathcal{V}}, N_{\mathcal{M}}, P) \tag{3}$$

Training Objective. The outputs from each decoder are used to define the objective function using Mean Squared Error (MSE) between the pixel values and latent representations extracted by the target encoder. We calculate only masked patches, following MAE [9]. Additionally, for latent representations, we include calculations for the `[CLS]` token along with the masked patches. The objective functions for pixel and latent decoders are respectively computed as follows:

$$L_{\text{pixel}} = \frac{1}{D_x \cdot |\mathcal{M}|} \sum_{i \in \mathcal{M}} \|\hat{x}_i - x_i\|_2^2, \quad L_{\text{latent}} = \frac{1}{D_t \cdot |\mathcal{M}|} \sum_{i \in \mathcal{M}} \|\hat{t}_i - t_i\|_2^2, \quad L_{\text{cls}} = \frac{1}{D_t} \|\hat{t}_0 - t_0\|_2^2 \tag{4}$$

Here, D_x and D_t are the dimensions of x_i and t_i, respectively. The overall objective function for training is defined as a sum of L_{pixel}, L_{latent} and L_{cls}:

$$L = L_{\text{pixel}} + L_{\text{latent}} + L_{\text{cls}} \tag{5}$$

3 Experiments

Dataset. To validate the effectiveness of our method, we conducted experiments in two categories, following [1]: (i) Image classification task which requires high-level semantics using ImageNet-1K, CIFAR10, CIFAR100, iNaturalist2021 and Places365 datasets. (ii) Object counting and depth prediction tasks which require low-level details using Clevr/Count and Clevr/Dist datasets. For iNaturalist2021, a mini version was used considering an appropriate data size. Since we performed pre-training on ImageNet-1K before conducting downstream tasks, we will refer to all datasets except ImageNet-1K as out-of-domain data.

Implementation Details. We pretrained the baselines and our method on ImageNet-1K and evaluated their performance through linear probing using the ViT-Base architecture. We compare our method against MAE [9] and I-JEPA [1], which are prominent methods for Pixel MIM and Latent MIM, respectively, and BootMAE [6], which combines these two methods. I-JEPA was pretrained for 600 epochs, MAE and BootMAE for 800 epochs, and PiLaMIM for both 600 and 800 epochs. During

Table 1: Performance comparison on high-level and low-level visual tasks. The highest performances are highlighted in bold.

		(a) High-level Task					(b) Low-level Task	
Method	**Epochs**	ImageNet-1K	CIFAR10	CIFAR100	iNat2021	Places365	Clevr/Count	Clevr/Dist
MAE [9]	800	61.1	86.7	66.0	25.8	44.2	76.6	65.0
I-JEPA [1]	600	67.7	86.3	66.1	27.7	45.3	71.6	62.0
BootMAE [6]	800	67.6	90.8	72.7	26.6	**48.8**	82.6	**68.7**
PiLaMIM	600	67.0	91.0	73.0	29.5	47.0	80.7	66.8
	800	**69.2**	**92.4**	**74.2**	**31.0**	47.6	**83.8**	67.8

linear probing, both I-JEPA and BootMAE used the average pooled patch token from the last layer of the encoder, following the settings of each paper. For MAE and PiLaMIM, we used the `[CLS]` token from the same layer. However, for Clevr/Dist dataset, we observed a large performance gap between the `[CLS]` token and the average pooled patch token, so we standardized the evaluation using the `[CLS]` token. These tokens were processed through a linear layer, preceded by batch normalization, and used for final performance evaluation. All linear probing was performed over 100 epochs, with the best performance reported afterwards. Detailed model architectures and training settings are described in A.3 and A.4.

Main Results Table 1 (a) shows the performance of the high-level task with image classification evaluated using top-1 accuracy. As previously known [1], I-JEPA, which targets latent representations, outperforms MAE on most datasets. However, PiLaMIM significantly surpasses I-JEPA across all datasets except for ImageNet-1K, even at 600 epochs, particularly excelling on out-of-domain data. Specifically, our method achieves 92.4% accuracy on CIFAR10 and 74.2% on CIFAR100, which are 6.1% and 8.1% higher than I-JEPA, respectively. Furthermore, PiLaMIM outperforms BootMAE across most datasets, indicating that the integration of the `[CLS]` token enhances the model's ability to capture high-level semantics. Meanwhile, Table 1 (b) presents the performance of the low-level tasks such as object counting and depth prediction. As in prior studies [1], MAE, which targets raw pixels, outperforms I-JEPA in both tasks. Notably, PiLaMIM outperforms MAE across all tasks, achieving a 7.2% increase to 83.8% on Clevr/Count and a 2.8% increase to 67.8% on Clevr/Dist. Moreover, PiLaMIM performs comparably to or slightly better than BootMAE, particularly achieving a higher accuracy on Clevr/Count, demonstrating its effectiveness in capturing low-level details. We performed an additional visual analysis using t-SNE [18] in A.2.

These results demonstrates that our method, which targets pixels and latent representations, effectively captures both high-level and low-level visual features. Furthermore, the significant performance improvements observed in both categories of experiments indicate the importance of integrating both levels of visual features, which complement each other and lead to a richer visual representation.

Ablation. We conducted an ablation study to explore the importance of the `[CLS]` token. Table 2 shows the linear probing performance of PiLaMIM, pre-trained for 800 epochs with and without the `[CLS]` token. Even without the `[CLS]` token, PiLaMIM still outperforms MAE and I-JEPA on CIFAR100 and Clevr/Count. However, when the `[CLS]` token is included, the performance improves by 1.9% on CIFAR100 and 1.2% on Clevr/Count. This indicates that simply utilizing the `[CLS]` token can boost performance in both high-level and low-level visual tasks, suggesting potential for further improvements with its expanded use.

Table 2: Ablation study for `[CLS]` token.

Method	CIFAR100	Clevr/Count
Ours w/o `[CLS]`	72.3	82.6
Ours w `[CLS]`	**74.2**	**83.8**

4 Conclusion

We proposed PiLaMIM, a unified framework that combines Pixel MIM and Latent MIM, to effectively capture both high-level and low-level visual features. Our experiments showed that PiLaMIM consistently outperformed existing methods across high-level and low-level visual tasks, demonstrating the benefits of integrating both types of visual features. Notably, the inclusion of the `[CLS]` token further improved the performance in both visual tasks, highlighting its importance in capturing both levels of visual features more effectively. These findings demonstrate that PiLaMIM is capable of extracting rich and robust visual representations, making it well-suited for a wide range of visual tasks.

5 Acknowledgements

This work was supported by the Institute for Information and communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00010, Development of Korean sign language translation service technology for the deaf in medical environment), and the Artificial intelligence industrial convergence cluster development project funded by the Ministry of Science and ICT (MSIT, Korea) & Gwangju Metropolitan City.

References

[1] M. Assran, Q. Duval, I. Misra, P. Bojanowski, P. Vincent, M. G. Rabbat, Y. LeCun, and N. Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 15619–15629. IEEE, 2023. doi: 10.1109/CVPR52729.2023.01499. URL `https://doi.org/10.1109/CVPR52729.2023.01499`.

[2] A. Baevski, W. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 1298–1312. PMLR, 2022. URL `https://proceedings.mlr.press/v162/baevski22a.html`.

[3] H. Bao, L. Dong, S. Piao, and F. Wei. Beit: BERT pre-training of image transformers. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=p-BhZSz59o4`.

[4] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9630–9640. IEEE, 2021. doi: 10.1109/ICCV48922.2021.00951. URL `https://doi.org/10.1109/ICCV48922.2021.00951`.

[5] X. Chen, M. Ding, X. Wang, Y. Xin, S. Mo, Y. Wang, S. Han, P. Luo, G. Zeng, and J. Wang. Context autoencoder for self-supervised representation learning. *CoRR*, abs/2202.03026, 2022. URL `https://arxiv.org/abs/2202.03026`.

[6] X. Dong, J. Bao, T. Zhang, D. Chen, W. Zhang, L. Yuan, D. Chen, F. Wen, and N. Yu. Bootstrapped masked autoencoders for vision bert pretraining. In *European Conference on Computer Vision*, pages 247–264. Springer, 2022.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

[8] P. Gao, T. Ma, H. Li, Z. Lin, J. Dai, and Y. Qiao. Convmae: Masked convolution meets masked autoencoders. *CoRR*, abs/2205.03892, 2022. doi: 10.48550/ARXIV.2205.03892. URL `https://doi.org/10.48550/arXiv.2205.03892`.

[9] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. B. Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 15979–15988. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01553. URL `https://doi.org/10.1109/CVPR52688.2022.01553`.

[10] G. V. Horn, O. M. Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. J. Belongie. The inaturalist species classification and detection dataset. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8769–8778. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00914. URL `http://openaccess.thecvf.com/content_cvpr_2018/html/Van_Horn_The_INaturalist_Species_CVPR_2018_paper.html`.

[11] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1988–1997. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.215. URL https://doi.org/10.1109/CVPR.2017.215.

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

[13] Y. Liu, S. Zhang, J. Chen, K. Chen, and D. Lin. Pixmim: Rethinking pixel reconstruction in masked image modeling. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=qyfz0QrkqP.

[14] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[15] Z. Peng, L. Dong, H. Bao, Q. Ye, and F. Wei. Beit v2: Masked image modeling with vector-quantized visual tokenizers. *CoRR*, abs/2208.06366, 2022. doi: 10.48550/ARXIV.2208.06366. URL https://doi.org/10.48550/arXiv.2208.06366.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015. doi: 10.1007/S11263-015-0816-Y. URL https://doi.org/10.1007/s11263-015-0816-y.

[17] Z. Tong, Y. Song, J. Wang, and L. Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/416f9cb3276121c42eebb86352a4354a-Abstract-Conference.html.

[18] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[19] C. Wei, H. Fan, S. Xie, C. Wu, A. L. Yuille, and C. Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 14648–14658. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01426. URL https://doi.org/10.1109/CVPR52688.2022.01426.

[20] Y. Wei, A. Gupta, and P. Morgado. Towards latent masked image modeling for self-supervised visual representation learning. *CoRR*, abs/2407.15837, 2024. doi: 10.48550/ARXIV.2407.15837. URL https://doi.org/10.48550/arXiv.2407.15837.

[21] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu. Simmim: a simple framework for masked image modeling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 9643–9653. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00943. URL https://doi.org/10.1109/CVPR52688.2022.00943.

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

[23] B. Zhou, À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 487–495, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/3fe94a002317b5f9259f82690aeea4cd-Abstract.html.

[24] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. L. Yuille, and T. Kong. Image BERT pre-training with online tokenizer. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL `https://openreview.net/forum?id=ydopy-e6Dg`.

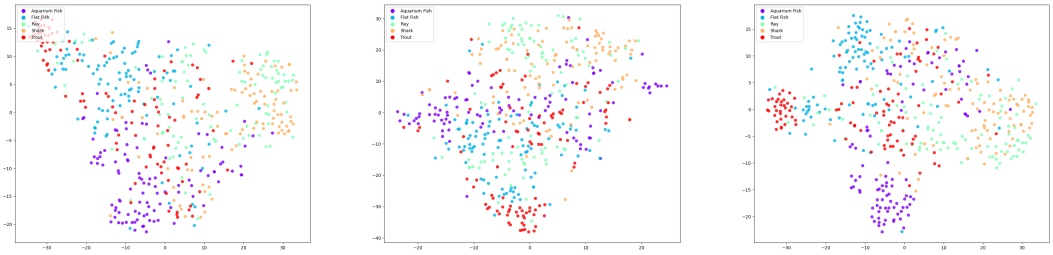
A Appendix

A.1 Limitations

Due to limited resources, we were unable to consider a broader range of visual tasks such as semantic segmentation and object detection. Additionally, we could only pre-train our method for up to 800 epochs, and were unable to explore the potential performance improvements with extended training. While MAE [9], I-JEPA [1], and PiLaMIM were all trained in our environment, BootMAE[1] [6] was used with a publicly available checkpoint for linear probing due to time constraints.

A.2 Visual Analysis

Figure 2 illustrates the t-SNE visualizations of images from the "Fish" superclass in the CIFAR100 validation set, using the encoders trained with MAE, I-JEPA, and PiLaMIM on ImageNet-1K. CIFAR100 is comprised of 20 superclasses, each containing 5 subclasses. Images within the same superclass share highly similar semantics, which increases the difficulty of accurate classification. Compared to MAE and I-JEPA, which target either pixel or latent representations, PiLaMIM shows better subclass clustering. This demonstrates that targeting both pixel and latent representation in MIM extracts richer and more robust visual representations.



(a) MAE (b) I-JEPA (c) PiLaMIM

Figure 2: The t-SNE visualizations of CIFAR100 "Fish" superclass images. Different colors represent different subclasses. They are better viewed by zooming in.

A.3 Architecture details

The encoder for all models consists of 12 ViT [7] blocks with 768 dimensions and 12 heads. The decoder configurations vary across models: MAE has 8 ViT blocks with 512 dimensions and 16 heads; I-JEPA and PiLaMIM have 6 ViT blocks with 384 dimensions and 12 heads; BootMAE has 2 cross-attention based ViT blocks with 512 dimensions and 12 heads.

The input image's height and width are 224, with a patch size of 16, resulting in a total of 196 patches being fed into the model. The masking ratio is 0.75, and no normalization was applied to the target pixels and latent representations. The λ used for updating the target encoder was initialized at 0.996 and linearly increased to 1.0 throughout pre-training.

A.4 Training details

Table 3 shows the pre-training settings on ImageNet-1K, while Table 4 presents the linear probing settings. For linear probing, we used a batch size of 16384 for ImageNet-1K [16] and Places365 [23], 8192 for iNat2021 [10], and 1024 for CIFAR10 [12], CIFAR100 [12], Clevr/Count [11], and

[1] `https://github.com/LightDXY/BootMAE`

Clevr/Dist [11]. We pre-trained PiLaMIM for 800 epochs using four NVIDIA A100 80GB GPUs over approximately 120 hours. Each linear probing downstream task was completed within 24 hours, using a single GPU for CIFAR10, CIFAR100, Clevr/Count, and Clevr/Dist, while 4 GPUs were used for the other datasets.

Table 3: Pre-training setting on ImageNet-1K.

config	value
optimizer	AdamW [14]
base learning rate	1.5e-4
weight decay	0.05
optimizer momentum	β_1 = 0.9, β_2 = 0.95
batch size	2048
learning rate schedule	cosine decay
warmup epochs	40
training epochs	800
augmentation	RandomResizedCrop

Table 4: Linear probing setting.

config	value
optimizer	LARS [22]
base learning rate	3
weight decay	0
optimizer momentum	0.9
batch size	varies by dataset
learning rate schedule	cosine decay
warmup epochs	10
training epochs	100
augmentation	RandomResizedCrop