Appendix:Self-supervised Representation Learning from Arbitrary Scenarios

Anonymous CVPR submission

Paper ID 8033

1. Evaluation metrics

Following common practices, we mainly use the top-1 accuracy to evaluate the semantic capacity of the pre-trained model for linear probing and fine-tuning classification task. Meanwhile, we adopt the box and mask mean average precision to validate the performance of transfer learning in object detection and instance segmentation tasks. Finally, we adopt the mean intersection of union to verify the transfer ability of the semantic segmentation task.

2. Pre-training settings

2.1. Large-scale settings

In experiments of COCO, both ImageNet-100 and COCO, and ImageNet-1K, for a fair comparison, we follow the settings of MAE [6]. We partition the image of 224×224 into 14×14 patches with the patch size being 16×16, and each patch as an image token. For ViT-Base model, it has 12 blocks, and each block has 768 feature dimensions and 12 self-attention heads. The batch size is set as 4096. Meanwhile, the weight decay, β_1 and β_2 for AdamW optimizer is set to be 0.05, 0.9 and 0.95, respectively. The warmup epochs is set as 40 epochs and the base learning rate *base_lr* $= 1.5e^{-4}$. In the experiment of ASL, the Transformer layer at the end of the encoder has 768 feature dimensions and 4 self-attention heads with 0.5 dropout ratio. For the ablation study and COCO pre-training experiments, we pre-train ASL with 800 epochs on COCO, then report these results of ImageNet linear probing and COCO detection. In pre-training experiments on both ImageNet-100 and COCO, we pre-train ASL with 800 epochs and 4000 epochs. In ImageNet-1K pre-training experiments, we pre-train the ASL with the same epochs of MAE.

2.2. Small-scale settings

In experiments of CIFAR-10 and CIFAR-100, we adopt the ViT-Small as the base architecture to verify the effectiveness of ASL in small-scale datasets. ViT-Small is pre-trained on CIFAR-10 and CIFAR-100 [10]. According to the prior work [3, 7], ViT-Small has 12 layers. For each layer, it has 384 feature dimensions and 6 self-attention heads. In our experiments, we adopt patch size 4×4 of image region as an image token and split the 32×32 images into 8×8 tokens. For the design of the decoder, its attention head and feature dimension are the same as the encoder. Besides, we set the decoder for MAE [6] to have the same depth, attention head, and dimension as ours. In the pre-training process, the batch size is set as 512, and weight decay is set as 0.05. The standard random cropping and horizontal flipping are used for data augmentation. Furthermore, we adopt AdamW optimizer [14], $\beta_1 = 0.9$ and $\beta_2 = 0.999$. *base_lr* $= 1e^{-3}$ to train the basic backbone, and the warmup epochs are set as 10 epochs. These ViT-Small models are pre-trained for 1600 epochs. In the experiment of ASL, the Transformer layer at the end of the encoder has 384 feature dimensions and 4 self-attention heads with 0.5 dropout ratio.

3. The downstream tasks settings of COCO, pre-training on both ImageNet-100 and COCO, and ImageNet-1K

3.1. The details of linear probing

For linear probing, we follow MAE [6] to evaluate the ImageNet pre-trained models, using the LARS [19] optimizer with momentum 0.9. The model is trained for 90 epochs. The batch size is 16384, the warmup epoch is 10 and the learning rate is 6.4. We adopt an extra BatchNorm layer [9] without affine transformation (`affine=False`) before the linear classifier. We set weight decay as zero. For ablation studies, we train 200 epochs and report the results of linear probing. The details are described in Table 1.

3.2. The details of end-to-end finetuning

Similarly, we adhere the hyper-parameters of MAE to end-to-end finetuning. The details are shown as Table 2.

3.3. The details of object detection and instance segmentation

By strictly following the training setting of MAE [6, 12], we train all models with the same simple formula: large-scale jitter [4], scale range ($[0.1, 2.0]$), AdamW ($\beta_1, \beta_2 =$

$0.9, 0.999$) with half-period cosine learning rate decay, linear warmup 0.25 epochs, and 0.1 drop path regularization. Moreover, the model is trained with 100 epochs and the batch size is set to be 64. Also, the learning rate is $8e-5$, and the weight decay is 0.1.

3.4. The details of semantic segmentation

Similarly, we fully follow the training setting of MAE. UperNet framework [18] is adopted as our segmentation method in our experiments. In particular, we use AdamW as the optimizer. The input resolution is set to be 512×512. The batch size is 16 and the layer-wise decay rate is 0.65. The model is end-to-end finetuned for 100 epochs.

4. The downstream tasks settings of CIFAR

In experiments of CIFAR-10 and CIFAR-100, the settings of downstream tasks are following as [7].

config	value
optimizer	LARS [19]
base_lr	0.1
weight decay	0
momentum	0.9
batch size	16384
learning rate schedule	cosine decay [13]
warmup epochs [5]	10
training epochs	90
augmentation	RandomResizedCrop

Table 1. **Linear probing setting.**

config	value
optimizer	AdamW [14]
base_lr	1e-3
weight decay	0.05
β_1,β_2 [1]	0.9, 0.999
layer-wise lr decay	0.75
batch size	1024
learning rate schedule	cosine decay
warmup epochs	5
training epochs	100
augmentation	RandAug (9, 0.5) [2]
label smoothing [16]	0.1
mixup [21]	0.8
cutmix [20]	1.0
drop path [8]	0.1

Table 2. **End-to-end finetuning setting.**

5. Compared with MAE pre-trained on ImageNet-1K

In order to assess the generalization capability of the ASL in arbitrary scenarios fairly and reasonably, we compared MAE pre-trained on the ImageNet-1K dataset with the ASL model pre-trained on a combination of ImageNet-100 and COCO, specifically examining its performance on the ImageNet-100 dataset. It is noteworthy that both the ImageNet-1K dataset and the mixed ImageNet-100 and COCO dataset share ImageNet-100 as a subset. Therefore, a more equitable and justifiable evaluation method for pre-trained models is to assess their performance on the ImageNet-100 and other datasets, in contrast to directly evaluating them on ImageNet-1K. As depicted in the Table 3, the performance of ASL, pre-trained for approximately 236k iterations, surpasses that of MAE trained for about 499k iterations, all the while utilizing only 70% of the computational load required by MAE. Moreover, the main text shows the performance of ASL outperforms that of MAE on COCO detection, instance segmentation, and ADE20k semantic segmentation. These results not only highlight the adaptability of ASL on arbitrary scenarios but also underscores its efficiency as a more effective algorithm.

6. The results of ViT-L

In order to demonstrate the generalization capability of ASL at a larger architecture, we conduct experiments using the ViT-L architecture, and the results are presented in the Table 4. The findings reveal that our approach achieves higher gains when employing a larger network structure.

7. The impact of global data augmentation in contrastive learning on MAE

In order to assess the impact of data augmentation previously validated in contrastive learning on MAE, we conducted some experiments in Table 5 using the ImageNet-1K dataset. All experiments were pre-trained for 200 epochs. The results indicate that the employed data augmentations are not conducive to improving MAE. These augmentations, implemented at a global level, prove impractical for MAE with patch-level learning. These experimental findings inspire us to propose patch-level feature enhancement as opposed to conventional global-level data augmentation for self-supervised learning.

8. Pre-training on OpenImages dataset

Here we provide 800-epoch results on OpenImages[11]. Its ImageNet-100 linear evaluation, object detection, and semantic segmentation are 87.6%, 51.7%, and 49.7%, outperforming the performance of MAE (83.5%, 49.9%, and 47.8%).

Method	Pre-train data	Iterations	Epochs	FLOPs	LP	FT
MAE [6]	ImageNet-1K	$\sim 249k$	800	$1\times$	80.5%	92.7%
MAE [6]	ImageNet-1K	$\sim 499k$	1600	$2\times$	85.3%	93.1%
ASL	ImageNet-100 + COCO	$\sim 236k$	4000	$\sim 1.4\times$	**85.9%**	**94.2%**

Table 3. **ImageNet-100 Top-1 accuracy of different methods under linear probing (LP) and fine-tuning (FT) setting.** We report top-1 accuracy on the ImageNet-100 `val` set. All of these methods adopt ViT-B.

Method	Pre-train data	Epochs	Arch.	LP	FT
ASL	ImageNet-100 + COCO	800	ViT-B	79.6%	92.4%
ASL	ImageNet-100 + COCO	800	ViT-L	85.1%	93.7%

Table 4. **ImageNet-100 Top-1 accuracy of different methods under linear probing (LP) and fine-tuning (FT) setting.** We report top-1 accuracy on the ImageNet-100 `val` set.

augmentation	Linear probing
baseline	58.8%
+ colorjitter	57.6%
+ grayscale	57.4%
+ gaussianblur	58.2%
+ solarize	55.8%

Table 5. **The impact of global data augmentation in contrastive learning on MAE.** We report top-1 accuracy on ImageNet-1K based on linear probing. All of these methods adopt ViT-B architecture.

9. Loss coefficient

The loss coefficient for $\mathcal{L}_{SEM}$ under AEE setting is set to 0.1, 0.5, 1, and 2, and the corresponding linear evaluation results of 47.0%, 48.0%, 48.6%, and 47.3%.

10. Runtime comparison between iBOT and ASL

Based on a batchsize of 32 for ViT-B, ASL achieves an iteration time of 0.2 s on V100 while iBoT is 1.6 s, despite iBoT having only 4 times FLOPs of ASL. According to our design, ASL's dual-branch features share the same model, enabling parallel computation for accelerated processing and allowing all features to be forwarded in a single pass. In contrast, hybrid methods like iBoT typically involve two models (student model and teacher model), leading to sequential computation. Specifically, after the forward computation of the student model is completed, the teacher model is then invoked, resulting not only in increased FLOPs but also longer processing times. We will add the time comparison.

11. Explanation of Tables in main text

Tables 1-4 constitute a comprehensive comparison which is divided into two parts. The first part (Tables 1 and 2) involves the comparison on the same dataset. The second part (Tables 3 and 4) comprises the comparison on the best performance, where we compare our ASL with other methods pre-trained on their favorite datasets according to their respective papers. The results show that ASL consistently achieves the SOTA in both cases. More discussions are also described in lines 417-445 (part 1) and 485-495 (part 2).

12. The detailed derivation

The MSE is equivalent to Eq(1), where C is a constant and equal to $p(x_{m_i})$.

$$
\begin{aligned}
\mathcal{L}_{single}(i) &= -\log p(x_{m_i}|x_{inputs},\theta,\xi) \\
&\cong -\log p(x_{m_i}|x_{inputs},\theta,\xi) + logC \\
&= -\log p(x_{m_i}|x_{inputs},\theta,\xi) + log1 + logC \\
&= -\log \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) + log1 + logC \\
&= -\log \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \\
&+ \log \int \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I})dx_{m_i} + \log C \\
&= -\log \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \\
&+ \log \int \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \cdot C dx_{m_i} \\
&= -\log \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \\
&+ \log \int \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \cdot p(x_{m_i}) dx_{m_i}
\end{aligned} \tag{1}
$$

For the second part of Eq(1), we utilize monte carlo method to solve $p(x_{m_i})$ and can obtain the Eq(2).

$$
\begin{aligned}
&\int \mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I}) \cdot p(x_{m_i}) dx_{m_i} \\
&= \mathop{}_{x_{m_i}\sim p(m_i)}[\mathcal{N}(x_{m_i};x_{p_i},\sigma^2_{noise}\mathbf{I})] \\
&\approx \frac{1}{N}\sum_{b=1}^{N} \mathcal{N}(x_{m_{i_{(b)}}};x_{p_i},\sigma^2_{noise}\mathbf{I})),
\end{aligned} \tag{2}
$$

The monte carlo method treats all pseudo labels in a training batch as random samples from $p(x_{m_i})$. Hence, for pseudo labels in a training batch $B = \{x_{m_{i_{(1)}}}, x_{m_{i_{(2)}}}, ...x_{m_{i_{(N)}}}\}$, the loss is defined as Eq(3), where $\lambda = 2\sigma^2_{noise}$ is a temperature coefficient.

$$\begin{aligned}\mathcal{L}_{single}(i) &= -\log \mathcal{N}(x_{m_i}; x_{p_i}, \sigma^2_{noise}\mathbf{I}) \\ &\quad + \log \int \mathcal{N}(x_{m_i}; x_{p_i}, \sigma^2_{noise}\mathbf{I}) \cdot p(x_{m_i}) dx_{m_i} \\ &\cong -\log \mathcal{N}(x_{m_i}; x_{p_i}, \sigma^2_{noise}\mathbf{I}) \\ &\quad + \log \int \sum_{b=1}^{N} \mathcal{N}(x_{m_{i_{(b)}}}; x_{p_i}, \sigma^2_{noise}\mathbf{I})) \\ &= -\log \frac{\exp(-||x_{p_i} - x_{m_i}||^2/\lambda)}{\sum_{x'_{m_i} \in B} \exp(-||x_{p_i} - x'_{m_i}||^2/\lambda)},\end{aligned} \tag{3}$$

13. The proof of maximizing likelihood estimation is equivalent to minimize MSE

For the likelihood estimation, it can be expressed as below with the function L where θ is the model, i is the index for the sample, and f is the probability density function:

$$L(\theta) = \prod_i f(x^i|\mu, \sigma^2) \tag{4}$$

The probability density function f for a Gaussian:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{5}$$

For maximum likelihood estimation, the y is the label, and y_p is the prediction by the model, the training stage of the model is modeled as the Gaussian distribution $y \sim \mathcal{N}(y_p, \sigma^2 I)$, that is, the prediction is considered as the mean of a noisy prediction distribution:

$$\begin{aligned} argmax \prod_i f(y^i|y_p^i, \sigma^2) &= argmax \prod_i f(y^i|y_p^i, \sigma^2) \\ &= argmax \prod_i f(y^i|y_p^i, \sigma^2) \\ &= argmax \prod_i \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y^i - y_p^i)^2}{2\sigma^2}} \\ &\cong log(argmax \prod_i \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y^i - y_p^i)^2}{2\sigma^2}} \end{aligned} \tag{6}$$

The log maximum likelihood estimation:

$$\begin{aligned} log(argmax \prod_i \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y^i - y_p^i)}{2\sigma^2}} &\propto argmax \sum_i -\frac{(y^i - y_p^i)}{2\sigma^2} \\ &\propto argmin \sum_i (y^i - y_p^i)^2 \end{aligned} \tag{7}$$

From the above, it can be observed that maximizing likelihood estimation is equivalent to minimize MSE.

14. Limitations

We have not extended ASL to larger datasets [15, 17, 22] and larger architectures (*e.g.*, ViT-H) due to the resource and time consumption.

References

[1] Mark Chen, Alec Radford, Rewon Child, Jeffrey K Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703, 2020. 2

[2] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*, 2019. 2

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1

[4] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2918–2928, 2021. 1

[5] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 2

[6] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 1, 3

[7] Tianyu Hua, Yonglong Tian, Sucheng Ren, Hang Zhao, and Leonid Sigal. Self-supervision through random segments with autoregressive coding (randsac). *arXiv preprint arXiv:2203.12054*, 2022. 1, 2

[8] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016. 2

[9] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 1

[10] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1

[11] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object

detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128(7):1956–1981, 2020. 2

[12] Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick. Benchmarking detection transfer learning with vision transformers. *arXiv preprint arXiv:2111.11429*, 2021. 1

[13] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 2

[14] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1, 2

[15] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 4

[16] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 2

[17] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 4

[18] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision*, pages 418–434, 2018. 2

[19] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017. 1, 2

[20] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 2

[21] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 2

[22] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017. 4