SINDER: Repairing the Singular Defects of DINOv2 — Appendix

Haoqi Wang[1], Tong Zhang[1], and Mathieu Salzmann[1,2]

[1] School of Computer and Communication Sciences, EPFL, Switzerland
[2] Swiss Data Science Center, Switzerland

A Details on PCA Visualization

Our PCA visualization in Figures 1 and 3 follows the procedure in [8]. Given the feature map of an image, containing $H \times W$ patch tokens, we first extract the three leading principal components of these tokens. Each token will become a dim-3 vector after the PCA. Then we scale each component to the range $0 - 255$ and interpret them as the RGB channels. The tokens are reshaped to resolution $H \times W$, and we get the resulting PCA visualization.

B Visualization of Clamping Singular Values

As a sanity check in Section 2, we clamp the singular value of the weights of linear layers in DINOv2 to a smaller value. The PCA visualization is shown in Figure 5. For each linear layer, we decompose the weight matrix using SVD (for example, $W = USV^T$) and then clamp the singular values S to be less than a threshold γ and get $\tilde{S}$. At last, we replace the weight matrix with $U\tilde{S}V$. In Figure 5, we compared with $\gamma = 2.0$, 1.5, and 1.3, respectively. We can see that as γ decreases, the norms of defective patches also decrease, and the number of defective tokens becomes less. However, when *gamma* is too small, the semantics of the feature maps seem corrupted. So we would prefer learned optimal singular values rather than trimming them according to some manually designed thresholds.

C Visualization of Learning Target

We visualize the learning target defined by Equation (7) in the last row of Figure 6. The feature maps of the 9th, 19th, 29th, and 39th layers for the villa image are illustrated. For demonstrative purposes, we show the learning targets for all pixels, which are visually smooth. However, in real fine-tuning, only pixels that are detected as defective contribute to the loss. The corresponding singular defects using mask threshold $\mu = 4$ are shown in the third row in Figure 6.

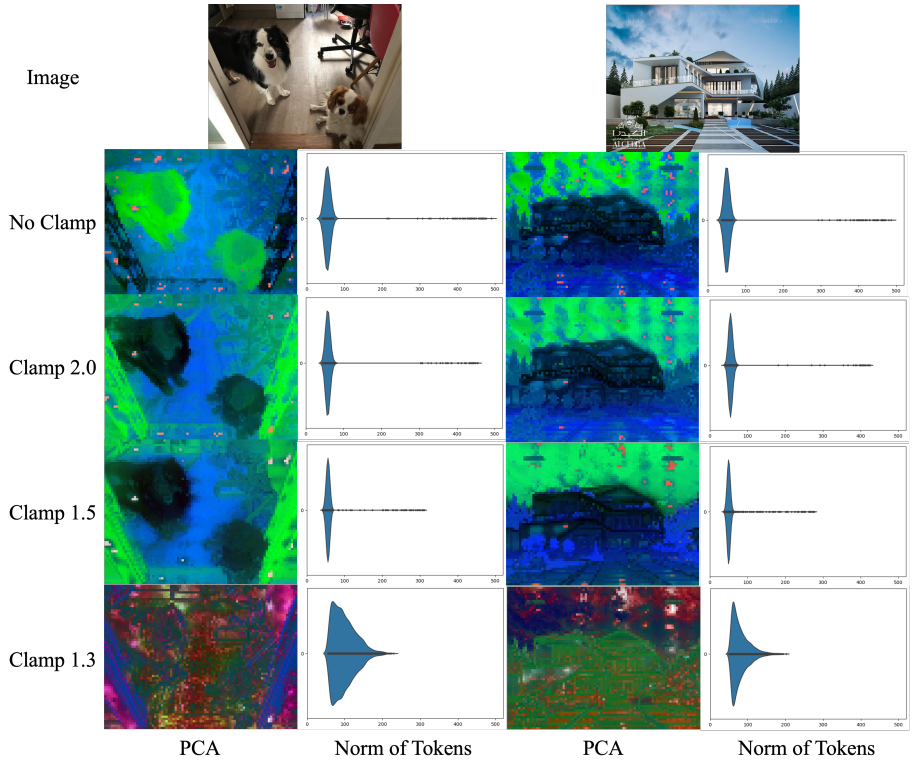


Fig. 5: Visualization after clamping the singular values of linear layers. The results of the two images are illustrated. The first and third columns are the PCA visualization of the feature map in the last layer. The second and fourth columns are the violin plots of the norm of the corresponding tokens.

D Visualization of Angles Between ν_i and Patch Tokens

In Figure 2d, the violin plot of the angles between the theoretical singular defect directions ν_i for layer-i and the patch tokens of the villa image are illustrated. To show that the isolated, anomalous points in the violin plot are indeed defective tokens, we present the corresponding PCA visualizations and the heatmap of angles in the first and second rows of Figure 6. In the angles heatmap, darker pixels mean that the angles between ν_i and the patch tokens are smaller.

E Visualization of the Learned Singular Values

We show the difference between the learned singular values and the original singular values in Figure 7. We observe that the differences are more striking for layers 5–25, and changes in other layers are modest. Generally speaking, the learned singular values are smaller than the original values.

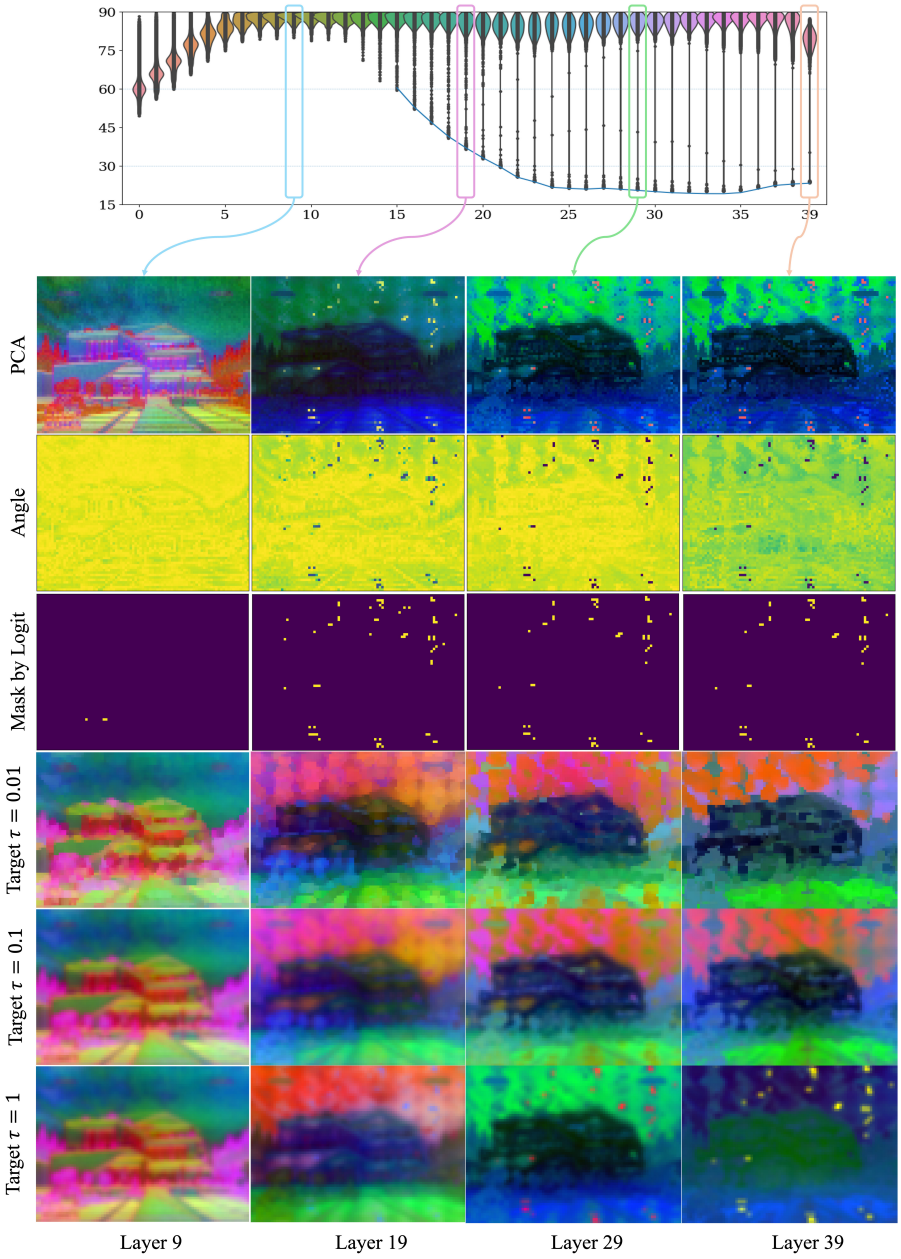
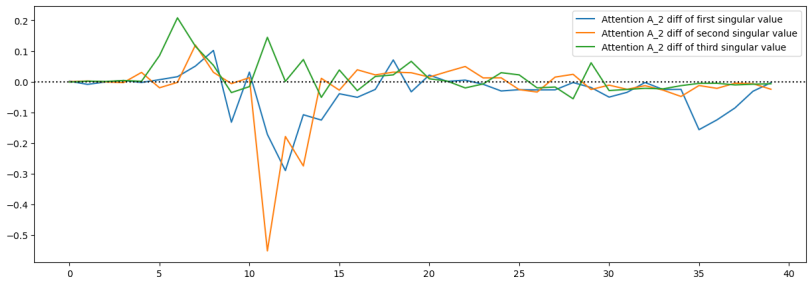
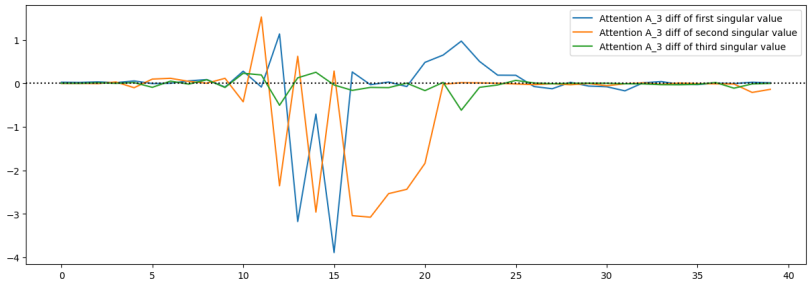


Fig. 6: The violin plot is the visualization of angles between the theoretical singular defect direction ν_i and patch tokens. The first row below the violin plot is the PCA visualization of patch tokens in the 9th, 19th, 29th, and 39th layers. The second row is the heat map of the angle between ν_i and patch tokens. The darker the color, the smaller the angles. The third row is the defective tokens detected by logits defined in Equation (5). The last three rows are the learning target under the temperature hyper-parameter $\tau = 0.01, 0.1, 1$. We use $\tau = 0.1$ in our experiments.



(a) Diff of Singular Values of A_2 in the Attention Block



(b) Diff of Singular Values of A_3 in the Attention Block

Fig. 7: Difference between the learned singular values and the original singular values. Changes in the remaining singular values are relatively small compared to the leading ones. So, only the first three leading singular values are shown to avoid cluttering the figure. The x-axis is the layer index.

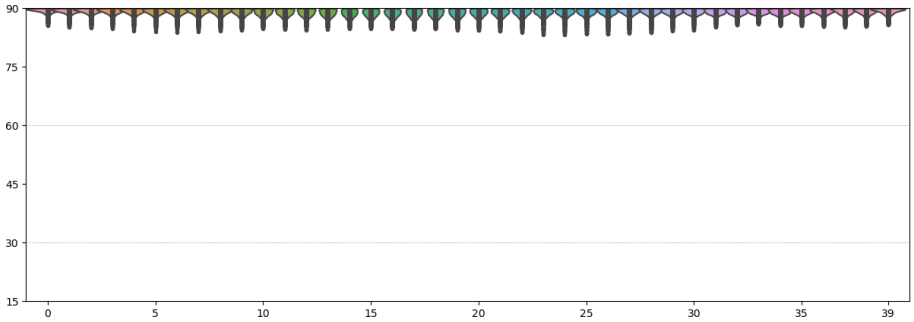


Fig. 8: The violin plot of angles between a random direction and the patch tokens of the villa image. The x-axis is the layer index.

F Visualization of Angles between Random Direction and Patch Tokens

To complement Figure 2, we show the violin plot of angles between a random direction and the patch tokens of the villa image in Figure 8. We can see that the pattern of the violin plot using a random direction is drastically different from the violin plots in Figure 2, which uses the leading left singular vectors. This shows that a random direction cannot be used to detect defective tokens.

G Detailed Configurations of Experiments

G.1 STEGO

Dataset We follow STEGO [5] to process the datasets, specifically, 27 classes of Cityscapes [2], and 3 classes of Potsdam-3 [6] are evaluated. We resize the images to 392×392 with center crop in training and 560×560 in testing. The training images are five-cropped.

Hyper-parameter We extended the STEGO's official codebase to support DINOv2 backbones. We use the hyper-parameters in STEGO's official repository, except for the backbone- and dataset-sensitive parameters, which are listed in Table 7. The hyper-parameters of STEGO used in the results of Table 1 are listed in Table 7.

G.2 CAUSE

Dataset We follow CAUSE [7] to process the datasets, specifically, 27 classes of Cityscapes, and 21 classes of PASCAL-VOC [4] are evaluated. Both training and testing resolution are 322×322.

Hyper-parameter We use the official codebase in CAUSE and adopt the default settings for all our experiments. specifically, we use the setting of CAUSE-TR.

Table 7: Hyper-parameters of STEGO.

Dataset	Backbone	neg inter weight	pos inter weight	pos intra weight	neg inter shift	pos inter shift	pos intra shift
Cityscapes	DINOv2	0.90	0.60	1.00	0.30	0.20	0.45
Cityscapes	DINOv2-Register	0.80	0.65	0.90	0.30	0.20	0.45
Cityscapes	DINOv2-SINDER	0.80	0.65	0.90	0.30	0.45	0.60
Potsdam-3	DINOv2	0.90	0.60	1.00	0.30	0.20	0.45
Potsdam-3	DINOv2-Register	0.90	0.60	1.00	0.30	0.20	0.45
Potsdam-3	DINOv2-SINDER	0.90	0.60	1.00	0.40	0.45	0.45

G.3 Classification KNN

We use the KNN implementation in the official codebase of DINOv2. The ImageNet-1K [3] dataset is used. The KNN performance on the validation set has been reported in Table 3. Specifically, $K = 10, 20, 100, 200$ are tested and the setting with the best top1 was reported.

G.4 Classification Linear Probe

We follow the linear probe implementation in the official codebase of DINOv2. The ImageNet-1K dataset is used. The linear probe performance on the validation set is reported in Table 3. The linear layer was trained for 10 epochs under learning rates of 1e-5, 2e-5, 5e-5, 0.0001, 0.0002, 0.0005, 0.001, 0.002, 0.005, 0.01, 0.02, 0.05, and 0.1, respectively, and the setting with the best top1 was reported.

G.5 Supervised Segmentation

We follow the evaluation protocol for supervised segmentation in DINOv2 and implement the training and testing using mmsegmentation [1]. Specifically, a linear layer is trained to predict classes from patch tokens. In the Linear setting, both training and testing images are resized to 512×512. For the Multiscale setting, they are rescaled to 640×640. Moreover, for the Multiscale setting, the patch tokens of the last four layers are concatenated, and the multiscale test-time augmentation was used in testing. For both ADE20k [9] and VOC2012, 40,000 iterations were trained.

References

1. Contributors, M.: MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation` (2020)
2. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
4. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
5. Hamilton, M., Zhang, Z., Hariharan, B., Snavely, N., Freeman, W.T.: Unsupervised semantic segmentation by distilling feature correspondences. arXiv preprint arXiv:2203.08414 (2022)
6. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9865–9874 (2019)

7. Kim, J., Lee, B.K., Ro, Y.M.: Causal unsupervised semantic segmentation. arXiv preprint arXiv:2310.07379 (2023)
8. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., HAZIZA, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. Transactions on Machine Learning Research (2023)
9. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 633–641 (2017)