A. Proofs

A.1. Proof of Proposition 4.1

Assuming the graphical model in Fig. 3 and the non-informative prior, we can describe the posterior $p(z|\mathbb{X},\theta) \propto p(\mathbb{X}|z,\theta)p(z)$ as the PoE of single-modal generative models;

$$p(z|\mathbb{X},\theta) \propto \prod_{j=1}^{M} p(x_j|z,\theta). \tag{34}$$

Then, the substitution of the Bayes' theorem relation $p(x_j|z,\theta) \propto p(z|x_j,\theta)p(x_j|\theta)$ into Eq. (34) derives Eq. (15), where $p(x_j|\theta)$ is absorbed into η_θ.

A.2. Proof of Claim. 4.3

Substituting Eqs. (19) and (20) into Eq. (11) yields;

$$\begin{aligned}\mathcal{J}_{\text{align}} &\propto \sum_i \mathbb{E}_{\delta(z-f_\phi(x_i))}[\eta_\theta \prod_j \log C_{\text{vMF}}(\kappa) e^{\kappa g_\theta(x_j)^\top z}] \\ &\overset{+}{\simeq} \sum_{i,j} \mathbb{E}_{\delta(z-f_\phi(x_i))}[g_\theta(x_j)^\top z] = \sum_{i,j} g_\theta(x_j)^\top f_\phi(x_i),\end{aligned} \tag{35}$$

where κ and $C_{\text{vMF}}(\kappa)$ are ignored as they are defined to be constants.

A.3. Proof of Claim. 4.5

Substituting Eqs. (16) and (15) into Eqs. (11) and (12) results in;

$$\mathcal{J}_{\text{align}} + \mathcal{J}_{\text{uniform}}$$

$$= \mathbb{E}_{q(z|\mathbb{X},\phi)}\left[\log \frac{p(z|\mathbb{X},\theta)}{\int p(z|\mathbb{X},\theta)p_{\mathcal{D}}(\mathbb{X})d\mathbb{X}}\right] \tag{36}$$

$$\overset{+}{\simeq} \mathbb{E}_{q(z|\mathbb{X},\phi)}\left[\log \frac{\prod_j p(z|x_j,\theta)}{\int \prod_j p(z|x_j,\theta)p_{\mathcal{D}}(x_j)d\mathbb{X}}\right] \tag{37}$$

$$= \mathbb{E}_{q(z|\mathbb{X},\phi)}\left[\log \prod_j \frac{p(z|x_j,\theta)}{\int p(z|x_j,\theta)p_{\mathcal{D}}(x_j)dx_j}\right] \tag{38}$$

$$= \sum_{i,j} \mathbb{E}_{q(z|x_i,\phi)}\left[\log \frac{p(z|x_j,\theta)}{\int p(z|x_j,\theta)p_{\mathcal{D}}(x_j)dx_j}\right]. \tag{39}$$

By applying Def. 4.2 and approximating the integral in Eq. (39) by the Monte Carlo method using a mini-batch $\mathcal{B}$, we can derive Eq. (26).

B. Implementation details

The implementation details of networks, pre-training, and linear evaluation are described in this section. Each experiment needs four NVIDIA V100 GPUs.

Experimental settings for networks; The architecture of the encoder f and the predictor h of VI-SimSiam are almost identical to those of DirectPred [50]. We use Resnet18 [21] as a backbone network of the encoder f, unlike the settings of DirectPred. We also add the kappa predictor to predict κ. The kappa predictor consists of an MLP, which has two fc layers. Its input layer has 512 dimensions and its output layer has 1 dimension (κ). BN and ReLU activation are applied to its input layer, while soft plus activation is applied to its output layer.

Experimental settings for pre-training; We use momentum-SGD for pretraining. The learning rates for VI-SimSiam, SimSiam, and DINO are 0.1, 0.2, and 0.03, respectively. The weight decay is 1e-4, and the SGD momentum is 0.9. The learning rate has a cosine decay schedule for SimSiam and DINO. A learning rate scheduler is not used for VI-SimSiam. The batch size is 512 for VI-SimSiam and SimSiam. It is 64 for DINO. The number of dimensions of the latent variable is 2048. In DINO, the temperature parameter for the teacher is 0.04, while that for the student is 0.1. The center momentum rate is 0.9. We do not use the momentum encoder as a teacher network in pre-training DINO.

We incorporate multi-crop [3] into augmentation. Multi-crop uses two types of views for training: standard and small-resolution views. Standard resolution views are generated by the same augmentation setting as that of SimSiam. SimSiam has five different augmentation types (*blur*, *color jitter*, *flip*, *grayscale* and *random crop*). Its strength is randomly determined with each applied augmentation. Fig 8 shows examples of augmented views. Low-resolution views are generated by the augmentation setting of SimSiam with modified random crop and resize parameters. In the augmentation setting of low-resolution views, the random crop scale is from 0.05 to 0.2, and the size is 96×96. We use two standard-resolution views and six low-resolution views per image in each experiment.

Experimental settings for linear-evaluation; We use LARS [60] as an optimizer for the linear evaluation. The learning rate is 1.6. The weight decay is 0.0, the SGD momentum is 0.9, and the batch size is 512. The image augmentation and preprocessing are referred to as those of SimSiam. We train 100 epochs and test the model with the highest Top-1 accuracy for the validation set. When we train it, we crop a random portion of an image and resize it to 224×224. The area of a random cropped image is from 0.08 to 1.0 of the area of the original image. When we validate and test it, we resize an image to 256×256 and crop the center 224×224.

ImageNet100 is split into only training and validation. Using this dataset, we set about 20% of the training split as a local validation split for tuning parameters.

Experimental settings for k-nearest neighbor classifi-

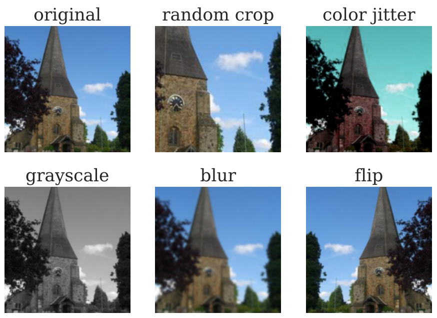

Figure 8: Examples of augmented images.

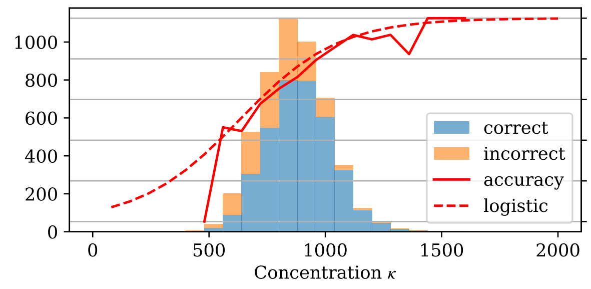

Figure 9: Histogram of the estimated concentration κ. The solid red line shows the Top-1 accuracy for each κ's range. The dotted line shows the logistic regression curve.

cation; We tuned the parameter k by calculating the top-1 accuracy of the local validation split for 200 ks. k is determined by Bayesian optimization each time.

C. Additional quantitative analysis of uncertainty

We investigate the effects of uncertainty on a classification task using methods other than AUROC. Fig. 9 shows κ histograms for correct and incorrect samples and plots each kappa range's Top-1 accuracy and the logistic regression curve. We use κ's predicted by the VI-SimSiam trained 500 epochs on ImageNet100 and labels predicted by linear classifiers with pre-trained features trained on 100 epochs. As the coefficient of κ in logistic regression is positive $(p < 0.01)$[6], the estimated κ increases, and thus, the greater the ease of estimating the class of the input image.

For each estimated label correctness, the mean and standard deviation of estimated κ is also given in Table 4. The mean of the κ of image features with the incorrect label is lower than those with the correct label.

[6]How to calculate the p-value is mentioned in Appx. J.

Table 4: Mean and standard derivation of κ for correct and incorrect samples.

Correct	Incorrect
890.52 ± 140.53	805.06 ± 131.20

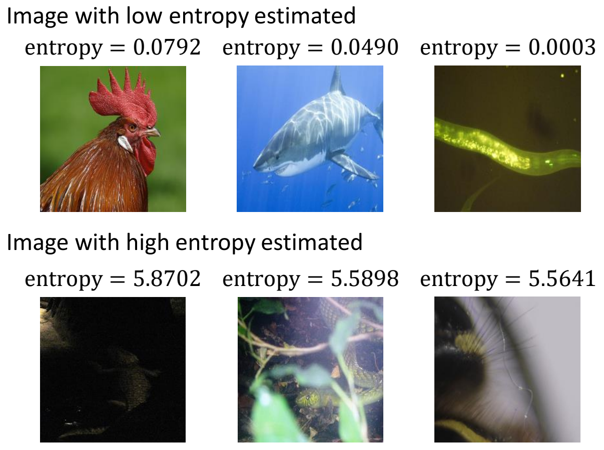

Figure 10: Input images and the entropy of the latent variable. The greater the entropy, the higher the uncertainty. The images with high estimated entropy, i.e., high uncertainty in the representations, appear to have less salient features than the others.

D. Evaluation of uncertainty estimation of DINO

We report that the entropy of the latent variable is related to the accuracy of the linear evaluation in Sec 6.4. In the current section, we qualitatively evaluate the uncertainty estimation of DINO by comparing entropies and images. Fig. 10 shows the images for which entropy is in the top and bottom 1%. When entropy is high, i.e., the uncertainty of the latent variable is predicted to be high, it is difficult to understand the features in the image. This result shows that DINO can learn the uncertainty of the latent variable, the same as VI-SimSiam.

E. Linear evaluation with κ

We attempt a new linear evaluation using the uncertainty parameter κ, which is related to the accuracy of linear evaluation. Therefore, we use the image with the highest κ from several randomly augmented images when testing. We use 30 augmented images. A random crop is used for augmentation. The Top-1 accuracy and the average of κ in the model for each epoch are shown in Table 5. This result shows a slight improvement in Top-1 accuracy. Although there is no significant change in the accuracy of the dataset used in this study, it can be effective in datasets where uncertainty can be reduced by image processing, such as noisy images.

Table 5: Top-1 accuracy and the average of κ Linear evaluation using the uncertainty parameter κ. "w/o κ" is a normal linear evaluation. "w κ" is a linear evaluation with κ which uses the image that predicted the highest κ from 30 augmented images in the test. When we evaluate the "w κ" setting, we test in triplicate and calculate their mean and standard deviation.

Method	100 epochs	200 epochs	500 epochs
	Top-1 accuracy		
w/o κ	73.32	76.74	77.48
w κ	73.51$\pm$0.43	77.36$\pm$0.04	77.61$\pm$0.17
	Average of κ		
w/o κ	867.71	811.47	848.23
w κ	917.81$\pm$0.36	852.15$\pm$0.42	893.12$\pm$0.24

Table 6: Top-1 accuracies of linear evaluation on Cifar10.

Method	100 epochs	200 epochs	500 epochs
SimSiam	77.43	85.95	91.96
VI-SimSiam	**89.92**	**93.02**	**92.75**

Table 7: Linear classification Top-1 accuracy in transfer learning. We use pre-trained 100 epochs of representations on ImageNet100 for each method. For the 200 and 500 epochs, the results of one experiment are presented, and for the 100 epochs, the mean and standard deviation of three experiments are given.

Method	epoch	Flowers	Food	DTD	Aircraft	SUN397	Pet	Cars
SimSiam [6]	100	14.8$\pm$0.5	46.7$\pm$1.6	35.7$\pm$7.1	10.4$\pm$1.3	39.5$\pm$1.8	40.5$\pm$2.5	8.5$\pm$1.5
VI-SimSiam	100	**64.5**$\pm$0.5	**48.4**$\pm$0.6	**51.8**$\pm$0.5	**16.2**$\pm$1.3	**46.4**$\pm$0.3	**50.0**$\pm$0.9	**13.7**$\pm$0.4
SimSiam	200	15.6	**55.8**	**55.8**	16.0	**51.2**	**54.5**	**16.2**
VI-SimSiam	200	**65.3**	49.4	53.7	**17.4**	47.4	50.8	14.7
SimSiam	500	28.0	**59.2**	**58.4**	17.7	**54.0**	**56.9**	**18.6**
VI-SimSiam	500	**64.8**	46.1	51.7	**17.9**	47.9	50.9	15.4

F. Linear evaluation on Cifar10 dataset

We conduct self-supervised pre-training with Cifar10 [30] dataset without labels to learn image representations using SimSiam and VI-SimSiam at 100, 200, and 500 epochs. Top-1 accuracy is the evaluation metric. Table 6 shows Top-1 accuracy on the validation split of ImageNet100. VI-SimSiam achieves a competitive result to SimSiam in all epochs.

G. Transfer learning

We evaluate the effectiveness of the representations for transfer learning on some image classification datasets (Flowers [38], Food [2], DTD [8], Aircraft [34], SUN397 [58, 57], Pet [43], and Cars [29]). We use representations by models pre-trained at 100 epochs on ImageNet100 dataset. The implementation details are presented in Appx. B. We report the Top-1 and Top-5 accuracy of VI-SimSiam and SimSiam in Table 7. Our method outperforms SimSiam on all datasets at 100 epochs and on some datasets at 200 and 500 epochs.

Table 8: Top-1 accuracy under semi-supervised learning on ImageNet100. We pretrain models on ImageNet100 without labels for each method. Then, we fine-tune them on 1 % or 10 % of ImageNet100 with labels. For the 200 and 500 epochs, the results of one experiment are presented, and for the 100 epochs, the mean and standard deviation of three experiments are given.

Method	epoch	1%	10%
scratch	100	10.8 ± 0.4	36.8 ± 0.6
SimSiam [6]	100	33.8 ± 1.5	64.6 ± 0.5
VI-SimSiam	100	$\mathbf{53.5 \pm 0.5}$	$\mathbf{68.8 \pm 0.0}$
SimSiam	200	49.7	**71.2**
VI-SimSiam	200	**56.2**	70.9
SimSiam	500	42.3	**73.3**
VI-SimSiam	500	**58.4**	72.0

Table 9: Top-1 accuracy of linear evaluation. For all methods, we pretrain a model of 200 epochs. We set a batch size to 512 for Simsiam and VI-SimSiam and 64 for DINO.

	SimSiam	VI-SimSiam	DINO
Top-1 accuracy	78.49	76.31	76.30

H. Semi-supervised learning

We evaluate the performance of the proposed method in a more realistic experimental setting, semi-supervised learning. Considering the excessive cost of labeling and the situation where only some data are labeled, we use labels for only 1% or 10% of the total ImageNet100. After pre-training networks without labels at 100 epochs, we fine-tune the entire networks at 200 epochs with labeled data, for a batch size of 128. Table 8 presents the Top-1 accuracy of each setting. Herein, VI-SimSiam consistently outperformed SimSiam for 1% of labels.

I. Linear evaluation of DINO

We demonstrate a linear evaluation of DINO. We use ImageNet100 dataset and set the number of epochs to 200. We set a batch size to 64 for DINO, unlike SimSiam and VI-SimSiam. Table 9 presents Top-1 accuracy with SimSiam, VI-SimSiam, and DINO.

J. Test of the relationship between two variables

This section describes whether a variable x is related to a variable y by testing the score a. Scores are assumed to be a relationship between two variables, e.g., coefficients from linear or logistic regression, and AUROC scores, among others.

First, a score is obtained from $X = \{x_1, x_2, ..., x_N\}$ and $Y = \{y_1, y_2, ..., y_N\}$. We then set the null hypothesis to H_0 and the alternative hypothesis to H_1. Set H_0 to mean that X and Y are unrelated and H_1 to mean that they are related. For example, if you want to test whether there is

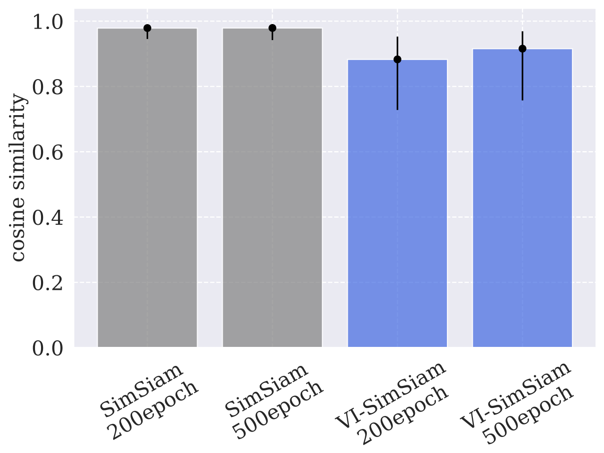

Figure 11: Variation of cosine similarity medians of 100000 random augmented image pairs. Error bars indicate the upper and lower quartiles.

a positive proportion, set $H_0 : a \leq 0$, $H_0 : a > 0$. Then, $Y'_j = \{y'_{j,1}, y'_{j,2}, ..., y'_{j,N}\}$ is obtained by random shuffling Y. This process creates a set of M dummy objective variables $Y'_1, Y'_2, ..., Y'_M$. M dummy scores $a'_1, a'_2, ..., a'_M$ are obtained from each Y'_j and X. We decide whether to reject H_0 or not by regarding M dummy scores as a test statistic.

K. Comparison of cosine similarity

In this section, we discuss the performance of the linear evaluation by comparing cosine similarity. When the cosine similarity is low during training, SimSiam only learns to increase the cosine similarity, while VI-SimSiam learns to decrease κ. The median of cosine similarity of 100000 random augmented image pairs is shown in Fig 11. Thus, the cosine similarity of VI-SimSiam is lower than that of SimSiam. In some cases, it is assumed that VI-SimSiam learns to minimize the loss by reducing κ rather than increasing the cosine similarity for inputs with representations that are difficult to predict.

L. Additional Experimental Results

In this section, we show another result of § 6.2. Fig. 12 shows twenty images estimated to have the highest κ, i.e., the lowest uncertainty. On the other hand, Fig. 13 shows twenty images estimated to have the lowest κ, i.e., the highest uncertainty.

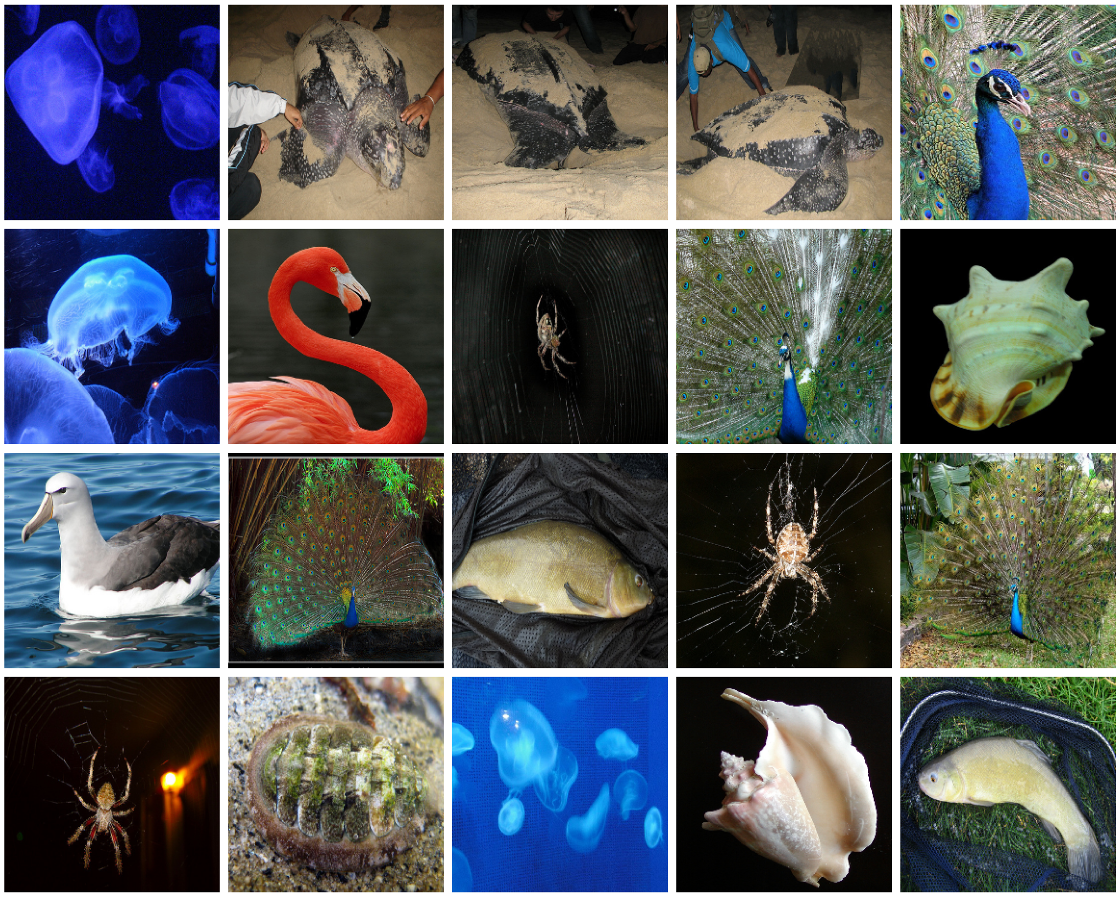

Figure 12: Twenty images estimated to have the highest κ.

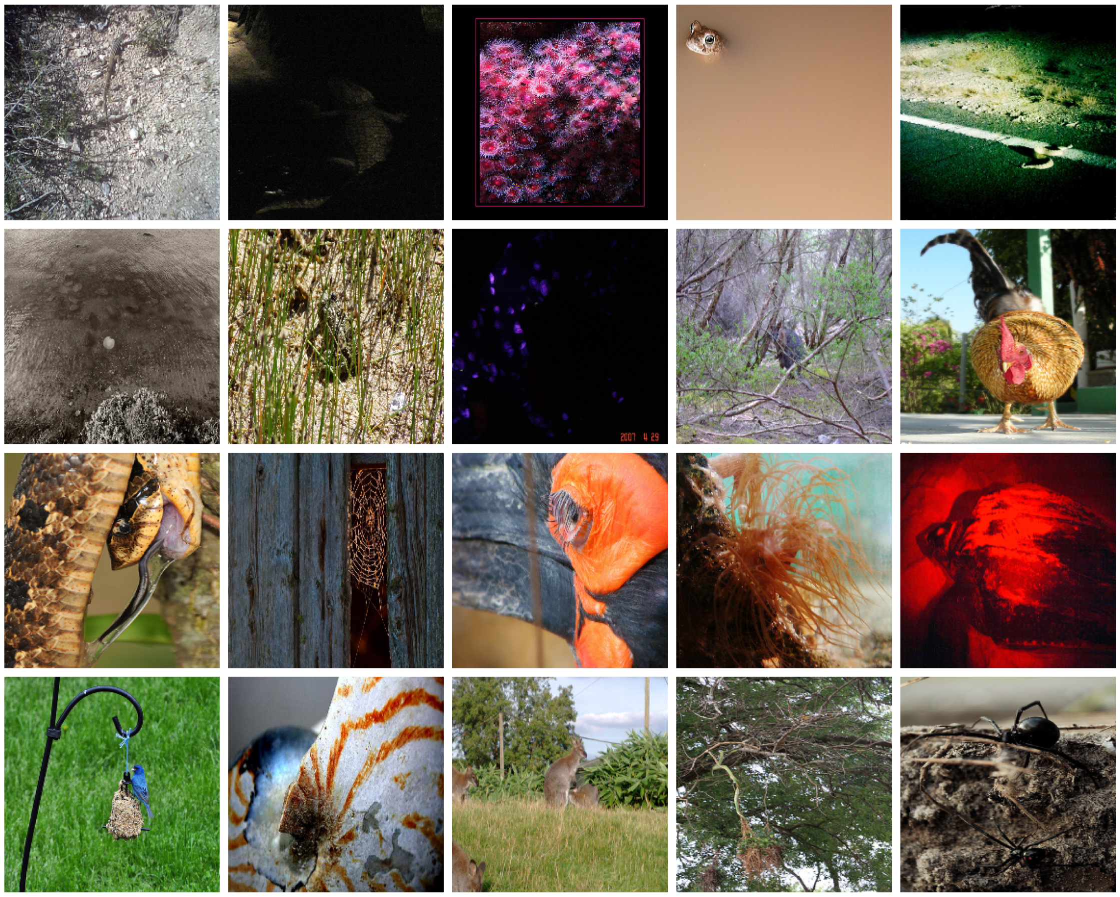

Figure 13: Twenty images estimated to have the lowest κ.