Focus Your Attention when Few-Shot Classification

Haoqing Wang, Shibo Jie, Zhi-Hong Deng
School of Intelligence Science and Technology, Peking University

Background

Traditional setting:

Meta-Training

Base dataset
(label cost, small size)

transfer

Few-shot Tasks

(support set, query set)

(support set, query set)

⋮

(support set, query set)

New setting:

Few-shot Tasks

self-supervised model
(unlabeled, massive data)

adapt

(support set, query set)

(support set, query set)

⋮

(support set, query set)

Fine-tuning pre-trained large models on few samples tends to overfit and is easy to be disturbed by noise information.

Analysis

➢ **Direct fine-tuning**

Model	**CUB**		**Cars**		**Places**		**Plantae**	
	1-shot	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**
MetaOptNet	57.0	85.1	24.1	57.9	50.0	71.1	36.9	63.7
VPT [26]	38.3	73.3	17.5	43.1	35.9	64.6	25.1	54.1
FT	25.0	66.7	15.3	42.2	23.6	55.0	22.0	51.4
LoRA [24]	55.3	83.6	22.5	54.7	48.4	68.2	34.2	62.1
SSF [34]	54.8	83.4	22.6	53.9	47.9	69.4	33.7	61.2

Table: accuracy on 20-way 1-shot/5-shot tasks; the pre-trained model is (ViT-B/16, DINO, ImageNet-1K)

➢ **Solution**: using a linear solver (e.g., MetaOptNet) to initialize the classification head

Model	**CUB**		**Cars**		**Places**		**Plantae**	
	1-shot	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**
NN	52.8	73.0	20.9	36.2	49.5	64.3	35.0	54.3
RR	56.6	84.0	24.0	56.2	49.9	68.7	36.8	62.0
SVM	52.8	78.7	20.9	37.3	49.5	70.8	35.0	57.7
ProtoNet [54]	52.8	79.8	20.9	39.3	49.5	71.2	35.0	57.7
R2D2 [5]	56.7	84.3	23.8	56.6	49.7	68.9	36.8	62.3
MetaOptNet [35]	57.0	85.1	24.1	57.9	50.0	71.1	36.9	63.7
Linear Probing	41.9	78.2	18.3	47.2	41.0	65.8	27.2	56.6
VPT [29]	52.9	81.1	23.3	54.5	48.0	69.6	33.9	60.2
FT	58.0	88.1	24.1	66.9	50.3	72.1	37.0	66.2
LoRA [27]	57.9	88.2	23.3	64.3	49.9	71.3	37.1	65.7
SSF [36]	57.8	88.4	23.8	62.3	50.2	73.4	37.2	66.0

Problem: fine-tuning sometimes cannot significantly improve performance beyond the classifier initialization (i.e., MetaOptNet) or even perform worse, especially for 1-shot tasks.

Methods

Focus on the key entities

- **Position prompts**: positions of the class-related key patches

1) for vision data; 2) non-limited backbone (columnar / pyramidal architectures) or pre-training way (single/multi-modal)

- **Text prompts**: only suitable for vision-language pre-trained models (multi-modal semantic alignment).

Target: using position prompts to guide the model to focus most attention on the key entities.

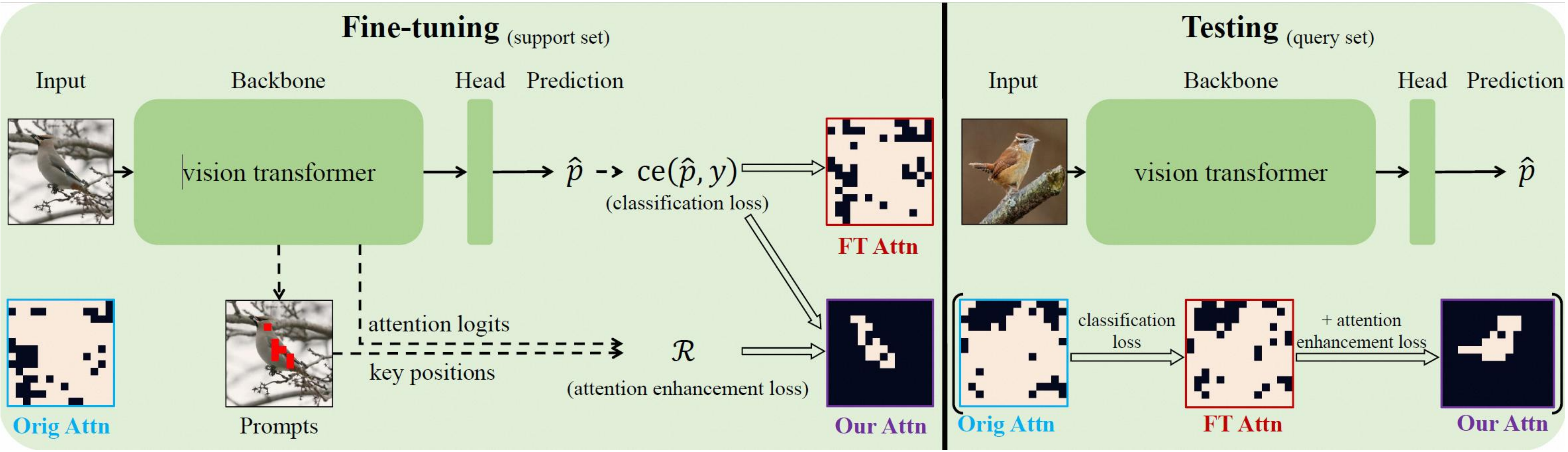

Figure: Focusing on key entities via *position prompts*. The original pre-trained model may attend to multiple entities in a single image, and the information from class-independent entities is actually noise in the current task. The few support samples make the fine-tuning with only classification loss unable to remove the noise information. We propose position prompts (red patches) to prompt the model where are the key patches of the input and focusing on them. This ability gained during fine-tuning can generalize from support samples to query ones. The white patches in attention visualization have the top highest attention scores and cover about 95% attention.

Methods

Locating position prompts

- Manually labeling (✕)
- Deep explanation method: **Grad-Rollout**

Let the input feature map of l-th layer be Z^l_{in}, its attention score is A^l. For sample (x, y), computing the gradient of the prediction score for class y, p_y, with respect to feature map Z^l_{in} as $\nabla_l = \partial p_y / \partial Z^l_{in}$, then the gradient term is

$$\mathbf{G} = \nabla_L \cdot \mathbf{V}_1 \in \mathbb{R}^{N\times 1}, \quad \mathbf{U}, \mathbf{S}, \mathbf{V} = \mathrm{svd}(\nabla_L)$$

Denoising: 1) only using the gradient at the top layer; 2) reserving its first principle component.

Introducing gradient term to attention score to achieve class-specific calculation, the final attention map of l-th layer is

$$\widehat{\mathbf{A}}^l = \mathrm{norm}\left(\mathbf{I} + \mathbf{A}^l + \lambda \cdot \mathbf{G}^T\right) \in \mathbb{R}^{N\times N}$$

Assuming the attentions are combined linearly along layers, the final importance scores are

$$\mathbf{s} = \mathrm{mean}(\widehat{\mathbf{A}}^1 \cdot \widehat{\mathbf{A}}^2 \cdot \ldots \cdot \widehat{\mathbf{A}}^L) \in \mathbb{R}^N$$

Methods

Attention enhancement

➢ Position prompts are used as the prediction target for attention, instead of in the input or middle phase.

$$\min_{\theta} \sum_{(x^s,y^s)\in\mathcal{T}_s} [\mathrm{ce}(f_\theta(x^s),y^s) - \alpha\cdot\mathcal{R}], \quad \mathcal{R} = \frac{1}{N\cdot|\Omega|}\sum_{n=1}^{N}\sum_{t\in\Omega}\ln\frac{\exp(\mathbf{q}_n\mathbf{k}_t^T/\tau)}{\sum_{m=1}^{N}\exp(\mathbf{q}_n\mathbf{k}_m^T/\tau)}$$

where Ω the index set of position prompts. $\mathcal{R}$ is calculated only on the last layer.
$\mathcal{R}$ aims to make the model to focus on the key entities. This ability generalizes from the support set to the query set.

Mark: our method introduces no new parametric modules which can not been learned using only few support samples.

➢ **Theoretical analysis:** increasing the information from the key patches and reduce that from other patches.

On the one hand, the InfoNCE estimate of $I(z,z_\Omega)$ is $\hat{I}_{InfoNCE}(z,z_\Omega) = \frac{1}{N\cdot|\Omega|}\sum_{n=1}^{N}\sum_{t\in\Omega}\left[\ln\frac{f(\mathbf{z}_n,\mathbf{z}_t)}{\frac{1}{N}\sum_{m=1}^{N}f(\mathbf{z}_n,\mathbf{z}_m)}\right] \le I(z,z_\Omega)$

so the regularization term $\mathcal{R}$ satisfy

$$\mathcal{R} = \hat{I}_{InfoNCE}(z,z_\Omega) - \ln N$$

which means $\mathcal{R}$ can increase the information from the key patches in all input tokens.

On the other hand, $\mathcal{R}$ can be factorized as

$$\mathcal{R} = \mathcal{R}^U + \mathcal{R}^A, \quad \mathcal{R}^U = -\frac{1}{N}\sum_{n=1}^{N}\ln\sum_{m=1}^{N}\exp(\mathbf{q}_n\mathbf{k}_m^T/\tau), \mathcal{R}^A = \frac{1}{N\cdot|\Omega|}\sum_{n=1}^{N}\sum_{t\in\Omega}\mathbf{q}_n\mathbf{k}_t^T/\tau$$

Increasing $\mathcal{R}^U$ can obtain more uniform distribution of input tokens, thus increasing $H(z)$;
Increasing $\mathcal{R}^A$ aligns the input tokens with the key patches and makes them more similar, thus increasing $-H(z|z_\Omega)$.

Experiments

1.1 Few-shot classification

➢ different backbone (Swin / ViT); different pre-training ways (single / multi-modal).

Model	CUB		Cars		Places		Plantae	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
NN	52.8	73.0	20.9	36.2	49.5	64.3	35.0	54.3
RR	56.6	84.0	24.0	56.2	49.9	68.7	36.8	62.0
SVM	52.8	78.7	20.9	37.3	49.5	70.8	35.0	57.7
ProtoNet [53]	52.8	79.8	20.9	39.3	49.5	71.2	35.0	57.7
R2D2 [5]	56.7	84.3	23.8	56.6	49.7	68.9	36.8	62.3
MetaOptNet [35]	57.0	85.1	24.1	57.9	50.0	71.1	36.9	63.7
Linear Probing	41.9	78.2	18.3	47.2	41.0	65.8	27.2	56.6
VPT [29]	52.9	81.1	23.3	54.5	48.0	69.6	33.9	60.2
FT	58.0	88.1	24.1	66.9	50.3	72.1	37.0	66.2
LoRA [27]	57.9	88.2	23.3	64.3	49.9	71.3	37.1	65.7
SSF [36]	57.8	88.4	23.8	62.3	50.2	73.4	37.2	66.0
FT + FORT	59.5 (1.5)	89.2 (1.1)	25.5 (1.4)	68.0 (1.1)	51.1 (0.8)	72.9 (0.8)	38.7 (1.7)	67.2 (1.0)
LoRA + FORT	62.5 (4.6)	89.5 (1.3)	26.8 (3.5)	65.7 (1.4)	50.8 (0.9)	72.4 (1.1)	38.5 (1.4)	66.9 (1.2)
SSF + FORT	62.3 (4.5)	89.6 (1.2)	26.5 (2.7)	64.2 (1.9)	51.3 (1.1)	74.4 (1.0)	39.0 (1.8)	67.5 (1.5)

Model	Aircraft		Pets	
	1-shot	5-shot	1-shot	5-shot
MetaOptNet [35]	25.5	55.5	71.9	89.6
FT	27.1	64.6	71.6	89.0
LoRA [27]	26.0	62.7	72.7	89.8
SSF [36]	26.5	61.6	72.7	90.0
FT + FORT	28.7	66.0	72.6	89.9
LoRA + FORT	31.3	63.9	74.8	90.9
SSF + FORT	30.9	63.1	74.4	91.2

Table: Accuracy on 20-way 1-shot / 5-shot tasks; the pre-trained model is (ViT-B/16, DINO, ImageNet-1K)

1.2 Few-shot classification

Model	CUB		Cars		Places		Plantae	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MetaOptNet [35]	49.1	81.1	21.0	51.1	48.9	70.4	36.0	63.1
FT	52.7	87.1	20.9	59.7	48.1	69.3	35.9	65.9
LoRA [27]	53.6	87.4	20.4	57.9	48.9	69.5	37.1	65.8
SSF [36]	54.9	87.6	21.4	57.2	48.7	70.8	35.9	64.7
FT + FORT	59.1 (6.4)	88.3 (1.2)	22.3 (1.4)	61.6 (1.9)	49.3 (1.2)	70.4 (1.1)	37.2 (1.3)	66.9 (1.0)
LoRA + FORT	59.9 (6.3)	88.5 (1.1)	21.9 (1.5)	59.4 (1.5)	49.7 (0.8)	70.4 (0.9)	38.7 (1.6)	66.9 (1.1)
SSF + FORT	60.1 (5.2)	88.9 (1.3)	23.4 (2.0)	58.8 (1.6)	49.8 (1.1)	72.0 (1.2)	37.6 (1.7)	66.0 (1.3)

Table: Accuracy on 20-way 1-shot / 5-shot tasks; the pre-trained model is (Swin-T/7, iBOT, ImageNet-1K)

Model	CUB		Cars		Places		Plantae	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MetaOptNet [35]	68.0	88.7	67.6	90.7	51.6	73.8	46.2	73.0
zero-shot [46]	84.1	84.1	88.0	88.0	76.6	76.6	61.2	61.2
CoOp [72]	84.4	90.4	91.3	94.6	77.3	81.1	63.8	76.2
Tip-Adapter-F [67]	86.9	92.0	92.2	95.3	79.8	82.0	68.3	79.3
PLOT++ [8]	87.4	92.0	92.2	95.5	79.9	82.7	67.7	78.8
LoRA [27]	86.3	92.6	92.3	95.8	79.8	84.1	67.4	80.0
LoRA + FORT	87.8 (1.5)	93.8 (1.2)	93.6 (1.3)	97.0 (1.2)	80.6 (0.8)	84.9 (0.8)	68.5 (1.1)	81.0 (1.0)

Table: Accuracy on 20-way 1-shot/5-shot tasks; the pre-trained model is (ViT-B/16, CLIP, WIT)

Experiments

2. Visualization of Position Prompts

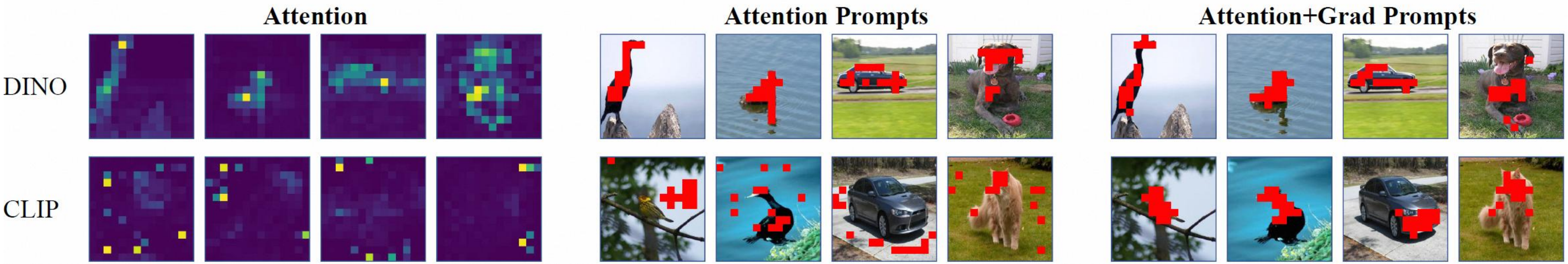

Figure: The attention map of [CLS] token in the pre-trained ViT-B/16 from DINO and CLIP, and the position prompts (red patches) obtained using the attention w/ or w/o gradient information.

3. Attention enhancement

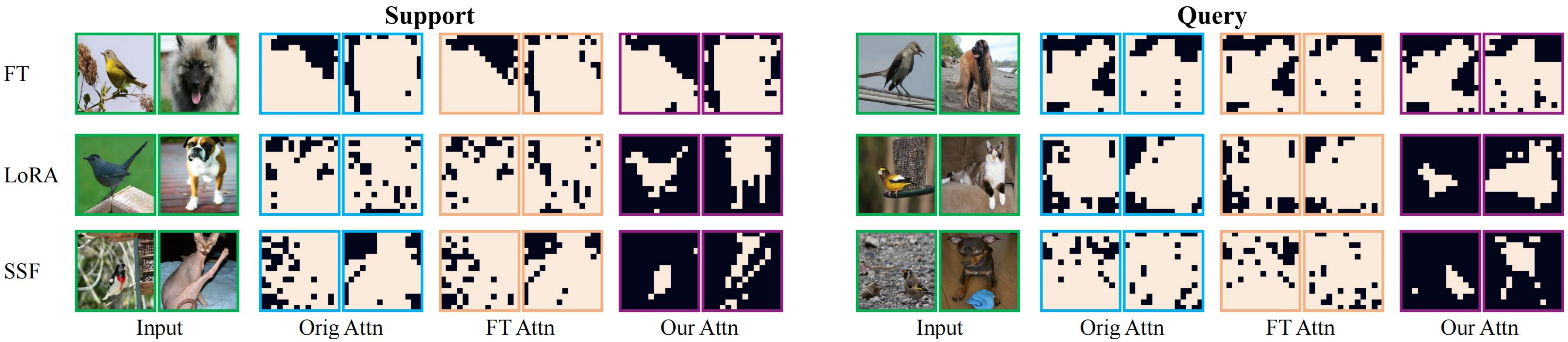

Figure: Visualization of the patches (white parts) with top highest attention scores and covering about 95% attention of [CLS] token in ViT-B/16. We use DINO pre-trained model for initialization