Incremental Few-Shot Semantic Segmentation via Multi-Level Switchable Visual Prompts

Supplementary Material

A. Implementation Details

In this section, we first elaborate on the process of region-unique prompt switching using `Gumbel_Softmax` and the design of the keys $\mathbf{K}_l$ of region-unique prompts. Secondly, we outline the settings of the proposed method, including the optimizer, learning rate, hyper-parameters, to ensure the reproducibility of the method. Thirdly, we describe the overall data flow and model structure. Finally, we explain the ways to perform incremental learning to expand the model. Due to the double-blind principle, **we will open source the code at our github after the review.**

A.1. Detailed Region-unique Prompt Switching Process

To enable end-to-end training, we employ a `Gumbel_Softmax` operation to switch the appropriate region-unique prompts. We first calculate the similarity between q_v and the keys of region-unique prompts, as

$$d_{k,j} = \frac{exp(q_{v,k}\mathbf{K}_{l,j} + \epsilon_k)}{\sum_{m=1}^{N} exp(q_{v,k}\mathbf{K}_{l,m} + \epsilon_m)}, \tag{A1}$$

where $\{\epsilon_k\}$ are i.i.d random samples drawn from the `Gumbel(0,1)` distribution. We compute the region-unique prompt to assign a centroid to by taking the one-hot operation of it `argmax` over all the keys. Since the one-hot assignment operation via `argmax` is not differentiable, we instead use the straight through trick in to compute the assignment matrix as

$$\hat{d} = \texttt{one-hot}(d_{\texttt{argmax}}) + d - \texttt{sg}(d), \tag{A2}$$

where `sg` is the stop gradient operator. With the straight through trick, $\hat{d}$ has the one-hot value of assignment to a single region-unique prompt, but its gradient is equal to the gradient of d, which makes the whole procedure differentiable and end-to-end trainable. After assigning q_v to keys of region-unique prompts, we can easily get the region-unique prompts response to q_v by merging all prompts, as:

$$p_{l,k}^{(i)} = \frac{\sum_{j=1}^{N} \hat{d}_{k,j}\mathbf{L}_j^{(i)}}{\sum_{j=1}^{N} \hat{d}_{k,j}}. \tag{A3}$$

This approach effectively solves the problem of gradient backpropagation by transforming the argmax process into a discrete variable sampling process.

Table A1. Ablation on the design of keys of region-unique prompts.

Method	1-shot		
	Base	Novel	HM
Rand	72.14	47.53	57.30
CoOP	73.04	49.08	58.71

A.2. The Design of the Keys of Region-unique prompts

As described in Sec. 4.2, to ensure stable training and provide meaningful initial keys for region-unique prompts, we leverage text embeddings generated by the CLIP text encoder following the CoOP[43] approach. Specifically, each region-unique prompt, which corresponds to a particular class, is assigned a key that aids in aligning the local features of input images. For each key, we use the CLIP text encoder to produce stable and meaningful embeddings by feeding it a prompt $\sigma = [V]_1[V]_2...[V]_n[CLASS]$, where $[V]_i$ represents vectors with the same dimension as word embeddings, n is a hyperparameter specifying the number of context tokens, and $[CLASS]$ denotes the class name's word embedding. By processing the prompt σ through the text encoder, we obtain a key tailored to each region-unique prompt. This method provides an effective initial value for the keys, mitigating the convergence issues often caused by random initialization, especially in few-shot scenarios. Moreover, it incorporates text modality knowledge, reducing the risk of overfitting to the limited samples in few-shot novel classes.

We further carry out an experiment to compare the performance with and without the keys generated by CoOP on VOC, as shown in Tab. A1. The table shows that with this key generation method, the model's ability to learn new classes is significantly enhanced.

A.3. Settings of the Proposed Method

Table A2. Settings of different datasets.

Dataset	L_g	M_0	L_s	h	ζ
VOC	24	8	8	2	0.7
COCO	40	15	16	4	0.7

During base training, the backbone and the query function are frozen, all prompts as well as the decoder are trainable. We use AdamW as optimizer with $\beta_1 = 0.9$,

$\beta_2 = 0.9$, weight decay 0.01, and a polynomial learning rate policy with a linear learning rate warmup. The model is trained for 20k iterations on VOC and 80k iterations on COCO with a learning rate of 2×10^{-4} and batch size of 8. During incremental training, we freeze all parameters but update the expanded stage-specific prompts and region-unique prompts. For both VOC and COCO, the model is trained for 400 iterations per step with a learning rate of 2×10^{-4} without the learning rate warmup. We compute the results via single-scale full-resolution images without any post-processing. The settings of model hyper-parameters on different datasets are shown in Tab. A2.

Additionally, the pre-defined background classes are `"sky", "wall", "tree", "wood", "grass", "road", "sea", "river", "mountain", "sands", "desk", "bed", "building", "cloud", "lamp", "door", "window", "wardrobe", "ceiling", "shelf", "curtain", "stair", "floor", "hill", "rail", "fence"`.

A.4. Detail Data Flow and Model Structure

(1) Prompts Generation: The images are input into pre-trained query function (DINOv2) to get global and local query features q_g and q_c which are used to match optimal prompts, formulated by Eq. (8), Eq. (9) and Eq. (13). (2) Image and Text Encoding: Image tokens concatenated with TP, SP and RP are input into each block of CLIP image encoder (ViT-B), which outputs the class token g and pixel features P. The names of each BG and FG classes are input into CLIP text encoder to get text embeddings T_{bg} and T_{fg}. (3) Pixel Decoding: As shown in Fig. A1, CMQE integrates the image global feature g with text embeddings T_{fg} and T_{bg} to generate class-specific queries Q_{fg} and Q_{bg} as formulated by Eq. (3). FG isolation separator and BG refinement separator share the same structures, composed of three cross-attention blocks. For each block, pixel-wise features P are input as the keys and values. Q_{fg} and Q_{bg} are input as the queries of the first block and the output of each block is input as the queries of the next block. The last block outputs the final masks (Eq. (4)).

A.5. Incremental Learning

We elaborate on the ways in which the model is extended during the incremental stage as follow. During incremental training stage, we need to expand three components: 1) text embeddings of novel classes, 2) slots of stage-specific prompts, 3) slots of region-unique prompts.

For expanding text embeddings, we just add novel classes to foreground text embeddings, which can be denotes as $T_{fg}^t \in \mathbb{R}^{(C_{fg}^{t-1}+N_e)\times D}$, where N_e denotes the number of novel classes, C_{fg}^{t-1} denotes the number of foreground classes of stage $t-1$.

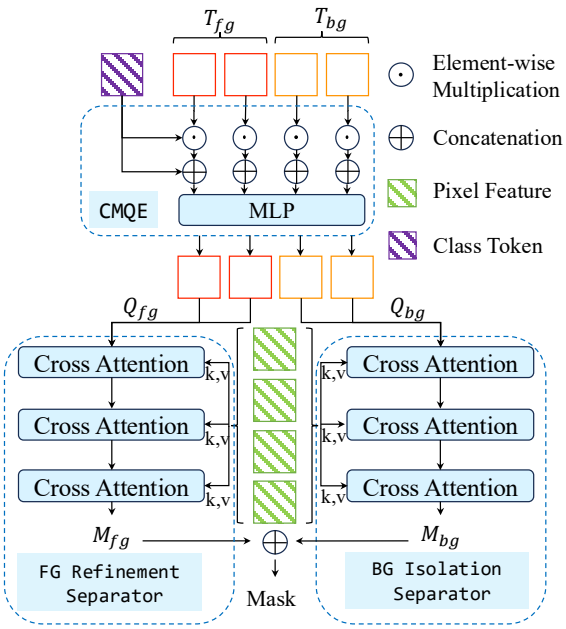

Figure A1. Detailed structures of the proposed decoder modules.

For expanding slots of stage-specific prompts, we concatenate the expanded $\mathbf{A}_e^{(i)} \in \mathbb{R}^{1\times D}, \mathbf{K}_e^{(i)} \in \mathbb{R}^{1\times D}, \mathbf{S}_e^{(i)} \in \mathbb{R}^{1\times L_s\times D}$ to matrix of stage $t-1$. Thereby, the stage-specific prompts integration of stage t can be formulated as:

$$\begin{aligned} \gamma =& \texttt{Softmax}(< q_g \odot [\mathbf{A}^{(i),t-1}; \mathbf{A}_e^{(i)}], \\ & [\mathbf{K}^{(i),t-1}; \mathbf{K}_e^{(i)}] > /\tau), \\ p_s^{(i),t} =& \gamma [\mathbf{S}^{(i),t-1}; \mathbf{S}_e^{(i)}] \in \mathbb{R}^{L_s\times D}, \end{aligned} \tag{A4}$$

where $\mathbf{A}^{(i),t-1}$,$\mathbf{K}^{(i),t-1}$,$\mathbf{S}^{(i),t-1}$ denotes the attention matrix, key matrix and stage-specific prompts of stage $t-1$ for layer i, $p_s^{(i),t}$ denotes the integrated stage-specific prompts.

For expanding slots of region-unique prompts, we concatenate the expanded $\mathbf{K}_{le}^{(i)} \in \mathbb{R}^{N_e\times D}, \mathbf{L}_e^{(i)} \in \mathbb{R}^{N_e\times D}$ with matrix of stage $t-1$. Thereby, the region-unique prompts integration of stage t can be formulated as:

$$\begin{aligned} p_l^{(i),t} =& \texttt{Gumbel\_Softmax}(q_v [\mathbf{K}_l^{(i),t-1}; \mathbf{K}_{le}^{(i)}]) \\ & [\mathbf{L}^{(i),t-1}; \mathbf{L}_e^{(i)}], \end{aligned} \tag{A5}$$

where N_e denotes the number of novel classes, $\mathbf{K}_l^{(i),t-1}, \mathbf{L}^{(i),t-1}$ denote the all keys and prompts of stage $t-1$, and $p_l^{(i),t}$ denotes the integrated region-unique prompts.

Note that due to the adoption of a switching mechanism, the number of visual prompts input to the model remain consistent at different stages, effectively preventing a significant increase in computational complexity.

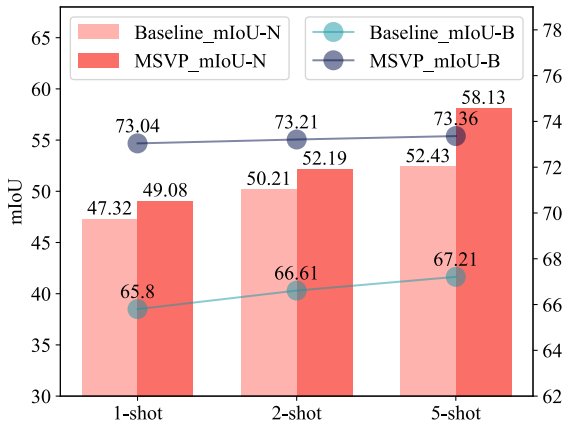

Figure A2. Performance comparison between the baseline and the proposed method on VOC under different shots.

B. More Ablation Studies

In this section, we first conduct experiments to demonstrate the effectiveness of MSVP compared to the baseline, and prove the effectiveness of orthogonality constraints. Secondly, we perform an ablation study on the scale and types of query functions. Thirdly, we prove the computation efficiency of MSVP. Finally, we carry out ablation experiments on the COCO dataset to further validate the proposed method.

B.1. Effectiveness of MSVP

To further demonstrate the effectiveness of the proposed MSVP, we compare the performance under different novel shots between the baseline and MSVP. As shown in Fig. A2, the performance of the baseline model does not increase as much as the proposed model with the growth of the shot. This demonstrates that MSVP effectively boosts the model's capacity to learn novel classes by efficiently switching prompts. In contrast, the baseline model suffers from diluted information as more visual prompts are added, leading to inadequate learning of novel classes. Meanwhile, the baseline's performance on base classes significantly outperforms that of the model with MSVP, and improves with more novel class samples. The proposed MSVP stores knowledge of different stages in independent visual prompts and switches them dynamically, thus avoiding knowledge of old stages corrupted.

B.2. Effectiveness of Orthogonality Constraints

We add orthogonality constraints to parameters of stage-specific prompts to avoid interference between existing and new knowledge and reduce catastrophic forgetting. In this section, we conduct an experiment on this loss to prove its effectiveness as in Tab. A3. The table shows that adding this loss function significantly enhances the performance of the base class by 5.69% mIoU, suggesting that it effectively minimizes interference from new knowledge on existing knowledge and helps prevent forgetting of previously learned classes.

Table A3. Ablation on $\mathcal{L}_{orth}$.

$\mathcal{L}_{orth}$	1-shot		
	Base	Novel	HM
wo	67.35	48.62	56.47
w	73.04	49.08	58.71

Table A4. Ablation on the scale of DINOv2.

Model	1-shot		
	Base	Novel	HM
ViT-S	71.62	48.57	57.88
ViT-B	73.04	49.08	58.71
ViT-L	72.68	48.76	58.36

Table A5. Ablation on different query functions.

Pre-train method	1-shot		
	Base	Novel	HM
MAE	50.44	39.00	43.98
BEiT	42.95	32.69	37.12
DINOv2	73.04	49.08	58.71

B.3. Ablation on Query Function

We choose pretrained ViT-B of DINOv2 as the query function to produce high quality global query features and local query features. In this section, we conduct an ablation study on the scale of the query function, as shown in Tab. A4. The model's performance improves significantly when the query function transitions from ViT-S to ViT-B. However, further scaling from ViT-B to ViT-L results in minimal performance gains, indicating that the larger model size does not substantially enhance effectiveness in our method.

We also perform experiments to evaluate different types of query functions, including MAE[44], BEiT[45], and DINOv2[15], as summarized in Tab. A5. For these experiments, we use ViT-B with various pre-training methods. The results show that DINOv2 achieves the best performance, likely due to its ability to provide both global and local features with high generalizability, thereby switching appropriate prompts accurately. In contrast, MAE and BEiT yield relatively inferior results, which may stem from their limitations in effectively capturing image-specific differences across different stages. Consequently, they struggle to generate tailored prompts that align with the unique characteristics of images at various stages.

Table A6. Ablation of Different Prompts on COCO.

SA	TP	SP	RP	1-shot SA	1-shot Novel	1-shot HM
✓				44.98	23.72	31.06
	✓	✓		48.26	24.65	32.63
	✓		✓	41.85	24.78	31.13
		✓	✓	44.24	20.54	28.05
	✓	✓	✓	**48.85**	**25.60**	**33.59**

Table A7. Ablation of the Framework on COCO.

Method	1-shot Base	1-shot Novel	1-shot HM
Ours	**48.85**	**25.60**	**33.59**
- FDD	48.18	23.13	31.25
- $\mathcal{L}_{ba}$	45.81	21.92	29.65
- CMQE	44.72	16.57	24.18

B.4. Ablation of Different Prompts on COCO.

We also carry out an experiment of different prompts on COCO to prove the effectiveness of our method, as shown in Tab. A6. When simply adding prompts as Eq. (2), the baseline achieves 44.98% mIoU-B and 23.72% mIoU-N, which has outperformed previous SOTA methods by a large margin. Replacing the vanilla prompt expanding strategy with the proposed task-persistent prompts and stage-specific prompts, the performance increases by 3.28% mIoU-B and 0.93% mIoU-N. That's because task-persistent prompts provide transferable knowledge across stages and stage-specific prompts extract relevant knowledge from other stages and enhance it with discriminative knowledge of the current stage, which offers a flexible way to switch knowledge of different stages, thereby achieving better abilities to keep old knowledge and learn new classes. Furthermore, with finer-grained region-unique prompts, performance on novel classes further rises to 25.60%, for the reason that region-unique prompts provide the model with knowledge of local details of specific classes. The table also shows that excluding task-persistent prompts leads to a performance decrease on both base and novel classes. It proves that general knowledge can not only help models maintain old abilities but also assist models in learning new abilities.

B.5. Ablation of the Framework on COCO.

We also carry out an experiment of the framework on COCO to prove the effectiveness of our method, as shown in Tab. A6. Replacing FDD with a single decoder results in a decrease in performance on novel classes. That's because FDD enables the model to process the salient features of novel classes and adapt to background changes separately, alleviating the confusion between novel classes and the background. Additionally, jointly optimizing the two separators with $\mathcal{L}_{ba}$ leads to a better performance, which mitigates the misaligned optimization directions caused by $\mathcal{L}_{van}$. It can also be concluded from the table that CMQE plays an important role in maintaining base knowledge and learning novel classes by generating generalizable class embeddings for base classes and novel classes.

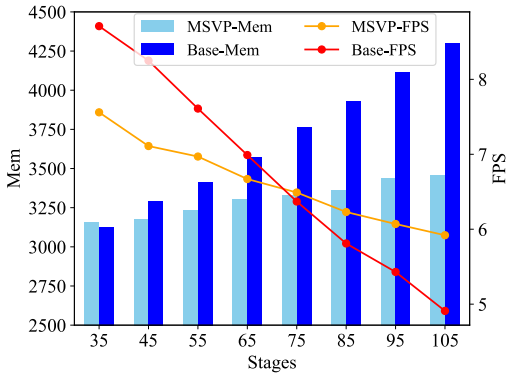

Figure A3. Comparison of MSVP and Base on FPS and Memory.

B.6. Computation Efficiency.

We conduct an experiment to compare FPS and memory usage between MSVP and Baseline as the incremental phase increases, on an NVIDIA A40 GPU using a 512×512 image. MSVP exhibits significant advantages in memory efficiency compared to the baseline. This improvement stems from our prompt-switching mechanism, which effectively maintains a constant input sequence length to the transformer model throughout progressive training stages, thereby avoiding memory accumulation. While the baseline shows marginally higher FPS during early incremental stages (<75 stages) due to the introduced query operation of MSVP, our method demonstrates superior computational sustainability as training progresses. Notably, the baseline suffers a sharp FPS degradation as its growing prompt inventory quadratically increases transformer's computational complexity ($O(n^2)$).

We further evaluated the FPS of PIFS, CaLNet, OINet, and our method on an NVIDIA H20 GPU using a 1024×1024 image. Results for the 30th stage are shown in Tab. A8, along with overall 1-shot performance on VOC. Our method achieves a good trade-off between accuracy and efficiency.

Table A8. Comparison on FPS and Parameters across Methods.

Method	PIFS	CaLNet	OINet	Ours
FPS	**27.01**	20.76	27.01	13.53
FT Para(M)	58.64	**0.001**	58.64	0.104
HM(mIoU)	26.7	28.2	28.3	**58.7**

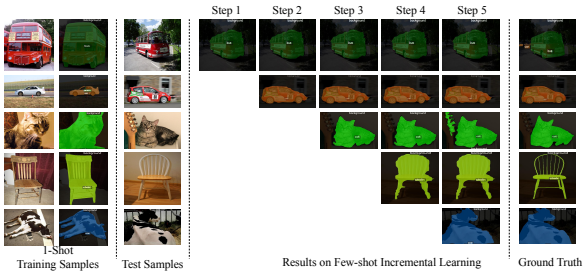

Figure A4. Step-by-step segmentation results of our method. Zoom in for better visualization.

C. Visualization

In Fig. A4, we visualize our step-by-step segmentation results for novel classes. The figure shows that our method effectively retains old class knowledge, enabling the model, even after multiple training rounds, to correctly predict old class samples. Additionally, our approach demonstrates strong novel class learning capability, as it can generalize to other test samples by learning from just one novel class sample. The visualization results effectively demonstrate the validity of the proposed prompt-based IFSS method.

D. Discussion

Our framework aligns with the Mixture of Experts (MoE) paradigm, where TP,SP and RP function as specialized experts for task-persistent, stage-specific, and region-unique knowledge. Crucially, the DINO-driven routing mechanism dynamically selects stage- and region-relevant experts, enabling parameter-efficient and context-aware computation. This design not only reduces redundancy but also embodies a conditional execution strategy—activating only necessary experts per input image, which resonates with broader trends in modular neural architectures for efficient inference.