Appendix of Hierarchical Prompt Learning for Scene Graph Generation

Xuhan Zhu [1,2]
zhuxuhan21@mails.ucas.ac.cn
Yifei Xing [1,2]
xingyifei22@mails.ucas.ac.cn
Ruiping Wang [*] [2,3]
wangruiping@ict.ac.cn
Yaowei Wang [1]
wangyw@pcl.ac.cn
Xiangyuan Lan [*] [1]
lanxy@pcl.ac.cn

[1] Pengcheng Laboratory, Shenzhen, China
[2] University of Chinese Academy of Sciences Beijing, China
[3] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

Contents

* denotes the corresponding author.

1 Further Implementation Details

The models are trained with the SGD optimizer, using the warmup and multistep learning rate adjustment strategies as in prior works [16, 30]. The batch size for sampled images is 16. On Visual Genome, models with Motifs or VCTree baselines [30] are trained for a maximum of 24K iterations, while models using PeNet baseline are trained for 30K iterations, as in the original PeNet. On OpenImage, the maximum iteration is 30K. Moreover, we describe in detail the selection of hyperparameters in Sec. 8.3. The positive gradients, negative gradients, and their gradient ratio can be obtained through Eq. 4 in Sec. 2.

For the visual encoder, besides the common visual encoder trained under a closed-set hypothesis (i.e., RX101-FPN [13, 26] backbone in Faster R-CNN), we also compare the CLIP-RN101 visual encoder [18] trained under an open-set setting in Sec.6.2. In addition to the CLIP language model, we also compare the BERT language model in Sec.8.2. In the case of CLIP, the language model is the text encoder of CLIP-RN101 [18]. For BERT, we utilize its base-scale model as the language model.

2 Gradient Analysis for Prompt Learning in SGG

In this section, we investigate the association between accumulated gradients and sample size in prompt learning for SGG. Additionally, we explore how positive and negative gradients

impact the posterior probabilities of SGG models. The probability for class r is given as:

$$\hat{p}_r = \frac{\exp(z_r/\tau)}{\sum_{j\neq r}\exp(z_j/\tau)+\exp(z_r/\tau)}, \tag{1}$$

where z_r is the logits of r-th class obtained through cosine similarities between text and relation embeddings, and τ is the temperature hyper-parameter. Given a training sample x_r with relation label r, the cross-entropy loss can be formulated as:

$$\mathcal{L}(x_r) = -log(\hat{p}_r). \tag{2}$$

The gradients of the $\mathcal{L}$ with respect to z for sample x_r are formulated as:

$$\begin{cases} \mathrm{g}_j^+ = \frac{\partial \mathcal{L}(x_r)}{\partial z_j} = -\frac{1}{\tau}(1-\hat{p}_j), & j = r \\ \mathrm{g}_j^- = \frac{\partial \mathcal{L}(x_r)}{\partial z_j} = \frac{1}{\tau}\hat{p}_j, & j \neq r, \end{cases} \tag{3}$$

where g_j^+ and g_j^- denote the generated positive and negative gradients during the optimization of sample x_r. Effectively, each label category r will get a positive gradient g_j^+, while other categories $j \neq r$ will receive a discouraging gradient g_j^- from sample x_r. We define the sample set with class r as $\mathtt{S}_r$ and with other classes as $\mathtt{S}_{\neg r}$. The accumulated positive gradients $\mathtt{G}_r^+$, negative gradients $\mathtt{G}_r^-$, and their ratio $\mathcal{A}_r$ for class r can be formulated as:

$$\begin{cases} \mathtt{G}_r^+ = \sum_{i\in \mathtt{S}_r} |(\mathrm{g}_r^+)_i| \\ \mathtt{G}_r^- = \sum_{i\in \mathtt{S}_{\neg r}} |(\mathrm{g}_r^-)_i| \\ \mathcal{A}_r = \frac{\mathtt{G}_r^+}{\mathtt{G}_r^-}, \end{cases} \tag{4}$$

where i denotes the index of sample $(x_r)_i$ in the sample set $\mathtt{S}_r$. Based on Eq. 4, we observe that the number of samples is proportional to the accumulated gradients. As the percentage of samples with class r (i.e., $\frac{|\mathtt{S}_r|}{|\mathtt{S}_r|+|\mathtt{S}_{\neg r}|}$) increases, there is a corresponding increase in the accumulated positive gradients for class r. On the other hand, an increase in the proportion of samples from other classes (i.e., $\frac{|\mathtt{S}_{\neg r}|}{|\mathtt{S}_r|+|\mathtt{S}_{\neg r}|}$) leads to a corresponding increase in the accumulated negative gradients for class r. Therefore, we arrive at the association between sample proportion and accumulated gradients as follows:

$$\begin{cases} \frac{|\mathtt{S}_r|}{|\mathtt{S}_r|+|\mathtt{S}_{\neg r}|} \propto \mathtt{G}_r^+ \\ \frac{|\mathtt{S}_{\neg r}|}{|\mathtt{S}_r|+|\mathtt{S}_{\neg r}|} \propto \mathtt{G}_r^-. \end{cases} \tag{5}$$

Therefore, obtaining a higher ratio of positive gradients for class r can increase the sample proportion of class r. Given that $\frac{|\mathtt{S}_r|}{|\mathtt{S}_r|+|S_{\neg r}|} = \frac{1}{1+|\mathtt{S}_{\neg r}|/|\mathtt{S}_r|}$, there are two ways to increase the sample proportion of class r: i) adding samples in $\mathtt{S}_r$ and ii) reducing samples in $\mathtt{S}_{\neg r}$.

Moreover, according to Bayes' theorem, the posterior probability $p(y|x)$ of y given image x can be formulated as:

$$p(y|x) \propto p(y)p(x|y). \tag{6}$$

For category r, its prior probability is defined as $p(y=r)$, denoted as p_r, which is proportional to the sample number $|\mathtt{S}_r|$ of class r [17, 23]. Therefore, based on Eq. 4-6, we can

derive the mathematical relationship as follows:

$$\begin{cases} \hat{p}_r \propto p_r \propto |\mathtt{S}_r| \propto \mathtt{G}_r^+ \\ \hat{p}_r \propto p_r \propto \frac{1}{|\mathtt{S}_{\neg r}|} \propto \frac{1}{\mathtt{G}_r^-} \\ \hat{p}_r \propto p_r \propto |\mathtt{S}_r| \propto \frac{1}{|\mathtt{S}_{\neg r}|} \propto \mathcal{A}_r. \end{cases} \tag{7}$$

In the proposed HP, the sample category scope is progressively narrowed for BP, FIP, and IIP, respectively. Moreover, to eliminate the influence of high-proportion discouraging negative gradients from majority classes, we respectively exclude the background class and head class relation embeddings in FIP and IIP, which can also retain as many relation embeddings for minority classes as possible. We define the training sample set for BP, FIP, and IIP as $\mathtt{S}^{\text{bp}}$, $\mathtt{S}^{\text{fip}}$, and $\mathtt{S}^{\text{iip}}$. For a tail class r, the relationship among them is: $|\mathtt{S}^{\text{bp}}_{\neg r}| > |\mathtt{S}^{\text{fip}}_{\neg r}| > |\mathtt{S}^{\text{iip}}_{\neg r}|$. Based on the conclusion in Eq. 7, we can derive the following relationship formula:

$$\mathcal{A}_r^{\text{bp}} < \mathcal{A}_r^{\text{fip}} < \mathcal{A}_r^{\text{iip}}. \tag{8}$$

Consequently, a tail class in IIP has a higher accumulated positive gradient ratio than in FIP and BP. Similarly, a foreground class in FIP has a higher accumulated positive gradient ratio than in BP, which is the plain prompt used in SGG models. Therefore, HP can aid in enhancing the learning of tail classes and foreground classes for SGG models. According to Eq. 7, we can infer that HP can also help improve their posterior probabilities.

3 Relation Feature Extractor

The relation feature extractor is used to extract regional relation features from input images I. More detailed schematics are shown in Fig. 1, which consists of the following modules:

Object Detector is used to detect the bounding box of nodes in scene graphs, which can be formalized as follows:

$$o = (b, c) = ([x, y, w, h], c) = \text{Obj}_{\text{Det}}(I), \tag{9}$$

where o means the nodes containing detected object boxes b=$[x, y, w, h]$ with classes c.

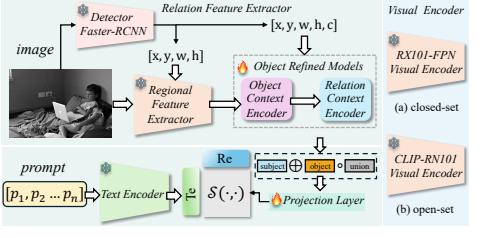

Figure 1: Overall framework of relation feature extractor. The right part (a) is the visual encoder RX101 from Faster-RCNN, and (b) is the visual encoder RN101 from CLIP. Some intermediate variables are omitted in this framework diagram for clarity.

Regional Feature Extractor is used to extract regional features of object regions and union regions (c.f. Eq. 10). Specifically, the visual encoder (denoted as Vis_{enc}) first extracts the image-level feature maps. Notably, in this work, in addition to the traditional Faster-RCNN visual encoder [19] obtained from closed-set training, we also introduce the visual encoder of CLIP [18] obtained from open-set training (c.f. Fig. 1). Subsequently, we use the RoIAlign operator (denoted as $\text{Roi}_{\text{align}}$) [8] to align the feature maps f to obtain the initial object features v and union features u, which can be formalized as follows:

$$\begin{cases} f = \text{Vis}_{\text{enc}}(I) \\ v = \text{Roi}_{\text{align}}(f, b) \\ u = \text{Roi}_{\text{align}}(f, b^u), \end{cases} \tag{10}$$

where b^u denotes the union regions of subject and object entities.

Object Refined Model is used to refine the object and union features to conclude the relation features. Specifically, it consists of the following two submoudles:

i) Object Context Encoder. Similar to the Faster-RCNN framework [19], we apply two MLP modules to encode object context. The object features encoded can be formalized as follows:

$$h = \mathrm{MLP}(v). \tag{11}$$

ii) Relation Context Encoder. The refined object features can be obtained using the relation context encoder (denoted as $\mathrm{RC}_{\mathrm{enc}}$, e.g., Motifs [28] and VCTree [24]). This encoder requires inputting bounding boxes o and object features, which can be formalized as follows:

$$\tilde{f} = \mathrm{RC}_{\mathrm{enc}}(h, o). \tag{12}$$

Next, we use two MLP modules to encode the union features. Subsequently, the union spatial encoder (denoted as $\mathrm{US}_{\mathrm{enc}}$), containing two convolutional layers [24, 25], is applied to obtain spatial union features for paired object bounding boxes (b^s, b^o). These spatial union features are then fused with the encoded union features to obtain the refined union features, which can be formalized as follows:

$$f_u = \mathrm{MLP}(u) + \mathrm{US}_{\mathrm{enc}}(b^s, b^o). \tag{13}$$

As described in the main paper, the final relation features are formalized as follows:

$$f_r = [\tilde{f}_s \oplus \tilde{f}_o] \circ f_u. \tag{14}$$

4 Efficient Discrete Triplet Prompt (EDTP)

As described in the main paper, the DTP faces a significant challenge of high computational complexity ($O(N_p \times |R|)$) as it requires sampling all triplets to establish the prompts. To mitigate the overburden on GPU memory, we optimize the engineering implementation by obtaining text embeddings for the triplet prompt before training. We refer to this process as the "offline process", and the DTP method optimized with this approach is denoted as **EDTP**. During the offline process, we use the text encoder to obtain text embeddings for all possible triplet prompts and store them for future use. During training, we obtain triplet word tokens for all object pairs and directly retrieve the corresponding triplet text embeddings from the pre-stored triplet text embeddings. Although this approach effectively solves the memory constraint issue, it still incurs significant computation and inference time. In addition to the extra overhead for downloading text embeddings and transferring them to the GPU memory, the EDTP must perform matrix multiplication between the text embeddings and visual embeddings for each triplet. Consequently, the computational complexity remains proportional to the large number of triplets processed.

5 Design Details for HP

5.1 Design Details for Informative Prompt

As mentioned in the main paper, HP has high flexibility to select any of the three basic prompts for utilization in HP. However, due to the inefficient computational efficiency of DTP, we limit our consideration to RP and CTP.

Algorithm 1 Pseudocode of Multi-Modal Contrastive Loss Function of Informative Prompts in a PyTorch-like style.

```
def ip_loss(t, f, t_ip, tau)
   # t: normalized text embeddings of triplets
   # f: normalized relation embeddings
   # t_ip: normalized text embeddings of informative prompts
   # tau: temperature coefficient
   # textual and visual matched cosine similarities
   logits_text_i = mm(t_ip, t.T) * exp(tau)
   logits_text_j = mm(t, t_ip.T) * exp(tau)
   logits_vis_i = mm(t_ip, f.T) * exp(tau)
   logits_vis_j = mm(f, t_ip.T) * exp(tau)
   # symmetric contrastive loss function
   labels = arange(t_ip.shape[0])
   l_t_i = CELoss(logits_text_i, labels, axis=0)
   l_t_j = CELoss(logits_text_j, labels, axis=0)
   l_v_i = CELoss(logits_vis_i, labels, axis=0)
   l_v_j = CELoss(logits_vis_j, labels, axis=0)
   loss = (l_t_i + l_t_j + l_v_i + l_v_j) / 4
   return loss
```

`mm`: matrix multiplication; `exp`: exponential function; `arange`: returns a tensor of equally spaced values within a given range; `CELoss`: cross-entropy loss function.

Subsequently, we will discuss which of RP and CTP is more suitable for use in HP. CTP introduces learnable variables to capture object information in comparison to RP. This endows it with two distinct advantages: **Firstly**, learnable parameters allow the model to capture information not only from the visual modality but also from the textual modality. The acquisition of multimodal information enables it to more effectively learn informative SGG. To be more specific, consider FIP as an example. As depicted in Eq. 15, we conclude two-modality similarities: z_r and $\tilde{z}_r$, which are learned in the visual and textual modality spaces, respectively. The textual modality knowledge comes from the triplet textual information.

$$\begin{cases} Visual: z_r = \mathcal{S}(t, e_{r\in R_+}), t = \mathcal{G}(\mathcal{T}_{\text{fip}}) \\ Textual: \tilde{z}_r = \mathcal{S}(t, tt_{r\in R_+}), tt = \mathcal{G}(\mathcal{T}_{\text{dtp}_{r\in R_+}}), \end{cases} \tag{15}$$

where tt represents the text embeddings of the input foreground triplets. The two formulas in Eq. 15 respectively indicate that the embeddings of the informative prompts need to be aligned with the visual embeddings and the triplet textual embeddings. Therefore, FIP is learned by incorporating information from both the visual modality space and the textual modality space. Moreover, we employ the contrastive learning system [18] to maximize the cosine similarities of matched pairs and minimize the cosine similarities of unmatched pairs. The multi-modal contrastive loss function is as follows:

$$\begin{cases} l_{\text{fip}-\text{v}}(e_{r\in R_+}) = -\log \frac{\exp(z_r/\tau)}{\sum_{j\in R_+, j\neq r}\exp(z_j/\tau)+\exp(z_r/\tau)} \\ l_{\text{fip}-\text{t}}(e_{r\in R_+}) = -\log \frac{\exp(\tilde{z}_r/\tau)}{\sum_{j\in R_+, j\neq r}\exp(\tilde{z}_j/\tau)+\exp(\tilde{z}_r/\tau)}. \end{cases} \tag{16}$$

In accordance with [18], we reverse the order of $e_{r\in R_+}$, t for z_r, and $tt_{r\in R_+}$, t for $\tilde{z}_r$ in Eq. 15, then we get the symmetric loss functions ($\bar{l}_{\text{fip}-\text{v}}$, $\bar{l}_{\text{fip}-\text{t}}$). Ultimately, the objective function of FIP is formalized as follows:

$$l_{\text{fip}} = (l_{\text{fip}-\text{v}} + l_{\text{fip}-\text{t}} + \bar{l}_{\text{fip}-\text{v}} + \bar{l}_{\text{fip}-\text{t}})/4. \tag{17}$$

When computing l_{iip}, replace $\mathcal{T}_{\text{fip}}$ with $\mathcal{T}_{\text{iip}}$, and restrict the training samples to the R_t set. The pseudocode for the multi-modal contrastive loss is visible in the Algorithm. 1. Subsequently, we validate our findings through experiments. As illustrated in Tab. 1, the experiments indicate that: 1) the performance of HP-i-CTP surpasses HP-i-RP when aligning

only a single visual modality. 2) The introduction of the textual modality information leads to noticeable performance gains; HP-i-CTP (w/ V+T) outperforms HP-i-CTP (w/ V) by 0.4, 0.6, and 0.5 points in R@100, mR@100, and MR@100, respectively. 3) Additionally, incorporating the multi-modal contrastive loss in HP can also enrich its learning with more valuable information, resulting in enhanced performance on mR@K and MR@K.

Secondly, in contrast to RP, the CTP allows the FIP, IIP, and MP in HP to have independent and diverse sets of parameters for utilization in the ensemble inference, which can aggregate the predictions from individual prompts and further increase the performance of HP. However, the fixed parameters in RP prevent it from possessing such capability. As shown in Tab. 1, after employing ensemble inference, HP-i-CTP outperforms HP-i-RP by 0.2, 1.8, and 1.0 points in R@100, mR@100, and MR@100, respectively.

Prompt Formats	R@50/100	mR@50/@100	MR@50/100
HP-i-RP(w/ V)	65.4 / 67.2	18.0 / 19.5	41.7 / 43.4
HP-i-CTP(w/ V)	65.9 / 67.6	18.1 / 19.5	42.0 / 43.6
HP-i-CTP(w/ V+T)	65.9 / 67.6	**18.7** / 20.1	**42.3** / 43.9
HP-i-CTP(w/ V+T) + ensemble	64.8 / 67.4	**18.7 / 21.3**	41.8 / **44.4**
HP(w/ V)	65.3 / 67.2	18.6 / 20.3	42.0 / 43.8
HP(w/ V+T)	65.7 / 67.4	19.1 / 20.6	42.4 / 44.0
HP(w/ V+T) + ensemble	64.2 / 66.0	**24.1 / 25.8**	**44.2 / 45.9**

Table 1: Ablation study of different prompt choices in HP-i and HP (on the PredCls task), w/ V indicates only learning the visual knowledge, while V+T means learning both textual and visual knowledge. Ensemble means the ensemble inference method.

Based on the above observations, we have chosen CTP as the informative prompts (including FIP and IIP) in the relevant experiments and exploited the multimodal constrative loss function to learn them. Moreover, the selection of the BP format is discussed for different tasks. Detailed explanations are provided in the experimental section of the main paper.

5.2 Design Details for CTP

When using CTP to learn novel predicates, there are some design details that need to be discussed. Due to the class-specific nature of CTP, where each set of learnable parameters is related to one predicate class seen in training, it is not applicable to the NPG task since novel classes are not given during the training process. To reduce this restriction, we include the same learnable parameters for all predicate classes as in [51]. Its formula is as follows:

Task	Methods	PredCls		
		R@ 50 / 100	mR@ 50 / 100	MR@ 50 / 100
NPG (Novel)	CTP (class-specific)	*N/A / N/A*	*N/A / N/A*	*N/A / N/A*
	CTP (unified)	1.2 / 1.2	1.4 / 1.4	1.3 / 1.3
Infor-SGG	HP (class-specific)	64.2 / 66.0	**24.1 / 25.8**	**44.2 / 45.9**
	HP (unified)	**64.3 / 66.3**	22.8 / 24.5	43.6 / 45.4

Table 2: Comparison of class-specific and unified CTP in NPG and Infor-SGG (informative SGG) tasks.

$$\mathcal{T} = \{[V^s, \texttt{REL}_i, V^o]\}_{0 \le i < |R|}, \tag{18}$$

where V^s and V^o denote the unified prompt that are the same vectors for any predicate class. As shown in Tab. 2, unified CTP is able to better predict novel classes in NPG. Therefore, in the NPG task experiments with CTP, we resort to using unified CTP.

Moreover, under the condition of sufficient training data, e.g., in the conventional informative SGG task, we compare the performance of both class-specific and unified CTP combined with the HP method. As shown in Tab. 2, we observe that the mR@K and MR@K performance of class-specific CTP are superior to that of unified CTP. This is likely attributed to the strength of class-specific parameters in learning multimodal knowledge, which proves advantageous for decision-making in different categories. Therefore, we prioritize class-specific CTP in the informative SGG task due to its enhanced performance.

Models	PredCls			SGCls			SGDet		
	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100
Motifs [28] *CVPR '18*	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6	38.9 / 39.8	8.3 / 8.8	23.6 / 24.3	32.8 / 37.2	6.8 / 7.9	19.8 / 22.6
+HP-i	64.8 / 67.4	18.7 / 21.3	41.8 / 44.4	40.0 / 40.7	11.1 / 11.8	25.6 / 26.3	32.9 / 37.4	8.0 / 9.3	**20.5 / 23.4**
+HP	64.2 / 66.0	**24.1 / 25.8**	**44.2 / 45.9**	39.3 / 40.2	**13.5 / 14.3**	**26.4 / 27.3**	31.9 / 36.3	**8.9 / 10.5**	20.4 / **23.4**
VCTree [24] *CVPR '19*	65.4 / 67.2	16.7 / 18.2	41.1 / 42.7	46.7 / 47.6	11.8 / 12.5	29.3 / 30.1	31.9 / 36.2	7.4 / 8.7	19.7 / 22.5
+HP-i	65.1 / 66.9	20.4 / 22.1	42.8 / 44.5	45.7 / 46.8	13.6 / 14.6	29.7 / 30.7	31.9 / 36.2	7.9 / 9.6	19.9 / 22.9
+HP	63.3 / 65.2	**23.8 / 25.7**	**43.6 / 45.5**	46.2 / 47.2	**14.3 / 15.7**	**30.3 / 31.5**	30.8 / 35.1	**9.2 / 10.8**	**20.0 / 23.0**
Transformer [25] *CVPR '20*	65.6 / 67.3	16.3 / 17.3	41.0 / 42.3	40.2 / 41.0	10.1 / 10.7	25.2 / 25.9	33.0 / 37.4	8.1 / 9.6	**20.6 / 23.5**
+HP-i	65.6 / 67.4	19.8 / 21.4	42.7 / 44.4	40.0 / 40.9	11.0 / 11.6	25.5 / 26.3	32.2 / 36.9	8.4 / 10.1	20.3 / **23.5**
+HP	64.0 / 65.9	**25.3 / 27.0**	**44.7 / 46.5**	39.3 / 40.1	**14.5 / 15.4**	**26.9 / 27.8**	31.7 / 36.1	**9.2 / 10.7**	20.5 / 23.4
GPSNet [15] *CVPR '20* ⋆	65.0 / 66.9	14.9 / 16.0	40.0 / 41.5	38.2 / 39.2	8.8 / 9.3	23.5 / 24.3	31.5 / 34.0	7.3 / 8.3	19.4 / 21.2
+HP-i	65.8 / 67.5	20.5 / 21.8	**43.2** / 44.7	37.9 / 39.0	10.7 / 11.3	24.3 / 25.2	31.3 / 33.9	8.9 / 10.3	**20.1 / 22.1**
+HP	61.7 / 63.8	**24.3 / 27.1**	43.0 / **45.5**	36.3 / 37.6	**13.1 / 14.3**	**24.7 / 26.0**	28.6 / 30.5	**10.4 / 12.0**	19.5 / 21.3
SHL [4] *CVPR '22* ⋆	65.1 / 66.9	16.0 / 17.3	40.6 / 42.1	39.7 / 40.5	9.6 / 10.2	24.7 / 25.4	32.2 / 36.7	7.3 / 8.6	19.8 / 22.7
+HP-i	64.9 / 66.7	20.1 / 22.5	42.5 / 44.6	39.0 / 39.7	11.6 / 12.3	25.3 / 26.0	32.5 / 36.9	7.9 / 9.3	**20.2** / 23.1
+HP	62.9 / 64.7	**25.0 / 27.1**	**44.0 / 45.9**	38.7 / 39.6	**12.7 / 13.6**	**25.7 / 26.6**	29.9 / 34.3	**9.9 / 12.0**	19.9 / **23.2**

Table 3: Comprehensive performance comparison of HP with different types of plain baseline models on VG. ⋆ means the results are reproduced following the open-source codes. The top-performing methods across all settings are underlined.

Models	PredCls			SGCls			SGDet		
	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100
Motifs	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6	38.9 / 39.8	8.3 / 8.8	23.6 / 24.3	32.8 / 37.2	6.8 / 7.9	19.8 / 22.6
Rwt [1, 28] *CVPR '19*	53.2 / 55.5	33.7 / 36.1	43.5 / 45.8	32.1 / 33.4	17.7 / 19.1	24.9 / 26.3	25.1 / 28.2	13.3 / 15.4	19.2 / 21.8
Rwt + HP	53.3 / 55.4	**37.3 / 39.3**	**45.3 / 47.4**	33.4 / 34.4	**21.6 / 22.8**	**27.5 / 28.6**	25.9 / 30.0	**15.4 / 18.2**	**20.7 / 24.1**
Rsp [28]⋆ *CVPR '21*	64.9 / 66.7	19.9 / 21.5	42.4 / 44.1	38.7 / 39.5	10.4 / 11.0	24.6 / 25.3	31.8 / 36.2	8.5 / 10.2	**20.2** / 23.2
Rsp + HP	62.1 / 64.2	**24.0 / 25.8**	**43.1 / 45.0**	36.9 / 37.9	**13.9 / 14.8**	**25.4 / 26.4**	30.3 / 34.7	**10.0 / 11.8**	**20.2 / 23.3**
GCL [4] *CVPR '22*	42.7 / 44.4	**36.1 / 38.2**	39.4 / 41.3	26.1 / 27.1	20.8 / 21.8	23.5 / 24.5	18.4 / 22.0	**16.8 / 19.3**	17.6 / 20.7
GCL + HP	50.4 / 52.5	34.5 / 37.0	**42.5 / 44.8**	26.8 / 27.8	**22.0 / 23.1**	**24.4 / 25.5**	21.3 / 24.9	16.3 / 18.8	**18.8 / 21.9**
Cacao [27]⋆ *ICCV '23*	34.5 / 35.7	**35.7 / 38.4**	35.1 / 37.1	24.5 / 25.1	**19.1** / 20.4	21.8 / 22.8	22.3 / 26.6	12.2 / 14.8	**17.3** / 20.7
Cacao + HP	43.3 / 49.5	31.5 / 37.8	**37.4 / 43.7**	32.3 / 34.3	19.3 / **21.8**	**25.8 / 28.1**	22.0 / 26.3	**12.5 / 15.2**	**17.3 / 20.8**

Table 4: Comprehensive performance comparison of HP with different types of debiasing baseline models on VG. ⋆ means the results are reproduced following the open-source codes.

6 Additional Experiments of HP

6.1 Additional Baselines

To better evaluate the effectiveness of our proposed HP on informative SGG, we comprehensively compare them on two types of baselines. Firstly, for plain baseline models without employing debiasing methods, we report the performance of the proposed HP using four different types of relation encoders. These encoders include the Motifs [28] model, which employs LSTM architecture; the VCTree [24] constructed by tree-based structure; the Tranformer [25] and SHL [4] models, which both incorporate transformer-based modules; and the GPSNet model [15] which utilizes the message passing module. Surprisingly, as shown in Tab. 3, our proposed HP-i and HP achieve better performance on almost all mR@K and MR@K than the baselines and maintain pretty good performance in R@K across all three tasks. Moreover, HP performs better than HP-i on mR@K and MR@K with various plain baseline models. These results demonstrate that HP is a plug-and-play, general method capable of enhancing the performance of informative predicates on plain baseline models.

Secondly, for debiasing baseline models, we report the performance of the proposed HP using four different types of debiasing models. These methods include the Cacao [27], which enhances the SGG dataset by generating more tail samples; the GCL method, which employs a set of classifiers specialized in learning different classes; the data resampling method (denoted as Rsp), which is a bi-level data resampling strategy [12] to balance the biased data distribution; and the reweighting method (denoted as Rwt), which amplifies loss weights for tail classes and diminishes loss weights for head classes. As shown in Tab. 4, we make two observations: 1) With regard to Cacao and GCL, their over-emphasis on tail classes at the expense of head classes significantly degrades their R@K despite slightly higher mR@K

Models	Vis_{enc}	PredCls			SGDet		
		R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100
RP	RX101-FPN	65.7 / 67.4	**16.7 / 18.0**	**41.2 / 42.7**	32.8 / 37.1	**7.6 / 9.0**	**20.2 / 23.1**
	CLIP-RN101	61.3 / 65.8	14.4 / 17.6	37.9 / 41.7	31.7 / 36.0	7.3 / 8.7	19.5 / 22.4
CTP	RX101-FPN	65.7 / 67.4	17.2 / 18.4	41.5 / 42.9	32.9 / 37.5	**7.7 / 9.1**	**20.3 / 23.3**
	CLIP-RN101	65.1 / 67.0	**17.6 / 19.1**	**41.4 / 43.1**	31.6 / 36.0	7.5 / 9.0	19.6 / 22.5
HP	RX101-FPN	64.2 / 66.0	24.1 / 25.8	44.2 / 45.9	31.9 / 36.3	8.9 / 10.5	20.4 / **23.4**
	CLIP-RN101	63.7 / 65.6	**24.4 / 26.3**	**44.1 / 46.0**	32.2 / 35.6	**9.4 / 10.9**	**20.8** / 23.3

Table 5: Comprehensive performance comparison of various visual encoders under the informative predicate learning task on VG. The baseline model is the Motifs model.

Split	Models	Vis_{enc}	PredCls			SGDet		
			R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100
Base	RP	RX101-FPN	64.5 / 66.4	14.6 / 15.8	39.6 / 41.1	31.9 / 36.2	6.4 / 7.5	**19.2 / 21.8**
		CLIP-RN101	63.5 / 65.5	**16.9 / 18.3**	**40.2 / 41.9**	30.6 / 35.0	**7.0 / 8.3**	18.8 / 21.7
	CTP	RX101-FPN	64.5 / 66.4	16.1 / 17.3	**40.3** / 41.8	29.5 / 33.8	7.1 / 8.5	18.3 / 21.2
		CLIP-RN101	63.6 / 65.7	**16.7 / 18.1**	40.2 / **41.9**	30.7 / 34.9	7.1 / 8.4	18.9 / 21.7
	HP	RX101-FPN	63.9 / 65.9	19.4 / 21.1	41.7 / 43.5	31.6 / 35.9	7.8 / 9.2	**19.7 / 22.5**
		CLIP-RN101	62.6 / 64.7	**22.0 / 23.8**	**42.3 / 44.3**	30.0 / 34.3	**8.6 / 10.2**	19.3 / 22.3
Novel	RP	RX101-FPN	11.1 / 11.2	5.9 / 5.9	8.5 / 8.5	5.4 / 5.8	3.1 / **3.5**	4.3 / **4.7**
		CLIP-RN101	9.9 / 10.0	**10.6 / 10.7**	**10.3 / 10.4**	5.6 / 5.9	**3.2** / 3.3	**4.4** / 4.6
	CTP	RX101-FPN	1.2 / 1.2	1.4 / 1.4	1.3 / 1.3	**0.3 / 0.4**	0.3 / 0.4	0.3 / 0.4
		CLIP-RN101	1.1 / 1.2	**1.6 / 1.8**	**1.4 / 1.5**	0.1 / 0.2	**0.3 / 0.4**	0.2 / 0.3
	HP	RX101-FPN	13.4 / 13.4	7.9 / 7.9	10.6 / 10.6	6.9 / 7.7	5.5 / 6.3	6.2 / 7.0
		CLIP-RN101	16.8 / 17.1	**10.7 / 10.9**	**13.8 / 14.0**	7.9 / 8.9	**5.6 / 6.4**	**6.8 / 7.7**
HM	RP	RX101-FPN	18.9 / 19.1	8.4 / 8.6	14.0 / 14.1	9.3 / 10.0	4.2 / **4.8**	7.0 / **7.7**
		CLIP-RN101	17.1 / 17.4	**13.0 / 13.5**	**16.4 / 16.7**	9.5 / 10.1	**4.4** / 4.7	**7.1** / 7.6
	CTP	RX101-FPN	2.3 / 2.3	2.5 / 2.5	2.5 / 2.5	0.6 / 0.8	**0.6 / 0.8**	**0.6 / 0.8**
		CLIP-RN101	2.2 / 2.4	**2.9 / 3.3**	**2.7 / 2.9**	0.2 / 0.4	**0.6 / 0.8**	0.4 / 0.6
	HP	RX101-FPN	22.1 / 22.3	11.2 / 11.5	16.9 / 17.1	11.3 / 12.7	6.5 / 7.5	9.4 / 10.7
		CLIP-RN101	26.5 / 27.1	**14.4 / 15.0**	**20.8 / 21.3**	12.5 / 14.1	**6.8 / 7.9**	**10.1 / 11.4**

Table 6: Comprehensive performance comparison of various visual encoders under the novel predicate generalization task on VG. The baseline model is the Motifs model.

performance (only on the PredCls task), thus losing precision in the prediction of common predicates. However, by adding our HP, they preserve head-class performance; the loss on R@K is less significant, while mR@K is also greatly improved over the Motifs baseline. This shows that our method can greatly encourage the learning of foreground and tail-class predicates while alleviating head-class degradation. 2) With respect to Rsp and Rwt, our HP outperforms them on mR@K and MR@K across all three tasks. All these results demonstrate the effectiveness of our method when combined with debiasing techniques.

In conclusion, our HP is applicable to various types of SGG baselines, highlighting its strong transferability and generalizability. The extensive experimental results consistently demonstrate the stable enhancement achieved by HP across these baselines, particularly in informative predicate prediction. Furthermore, when integrated with debiasing methods, our HP attains a new state-of-the-art performance on mR@K and MR@K in informative SGG.

6.2 Additional Visual Encoder

As discussed in Sec. 3, we introduce two visual encoders. The first is the visual backbone from closed-set trained Faster R-CNN: RX101-FPN. The second is the visual encoder from open-set trained CLIP: CLIP-RN101. The object context model of pre-trained Faster-RCNN has a significant impact on the performance of SGCls mode, but this model is not available with the CLIP visual encoder. Therefore, to ensure a relatively fair comparison with RX101-

FPN, we omit the SGCls protocol. We conduct comparative experiments for these two visual encoders under informative predicate learning and novel predicate generalization tasks.

Firstly, as shown in Tab. 5, different prompts on the two visual encoders achieve close performance in terms of mR@K and MR@K metrics. Secondly, different prompts on CLIP-RN101 consistently outperform RX101-FPN across almost all mR@K and MR@K metrics in the novel split, as shown in Tab. 6. CLIP-RN101 also performs better on the HM metric with various prompts. Based on these results, we draw two conclusions: 1) Under a closed-set experimental condition where only base categories need to be predicted, both visual encoders perform similarly (c.f. Tab. 5). 2) Under an open-set experimental condition where novel categories need to be predicted, the CLIP visual encoder yields better performance in prompt learning (c.f. Tab. 6). Notably, we apply RX101-FPN as the visual encoder in the main paper to ensure fair comparison with previous works [24, 25, 28].

Additionally, HP outperforms RP and CTP in terms of mR@K and MR@K performance with both two visual encoders, which further demonstrates the effectiveness of HP.

6.3 Additional Method Comparison

In this subsection, we will conduct a comparative analysis to highlight the superiority of our HP compared to comparators. These methods primarily include:

6.3.1 Gradient-adjusting Methods.

HP involves adjusting propagated gradients. We compare it against other gradient-adjusting approaches, including EQL V1 and V2 [21, 22]. The differences between them include: **1) Forms.** EQL V1 and V2 [21, 22] adjust gradients by relying on the loss functions, whereas HP adjusts gradients based on the gradient distribution as a prior and employs specialized prompts to adjust the gradient ratios of foreground and tail classes. This makes HP a model-level enhancement rather than a loss-level modification. Thus, these two types of methods are different in form and not conflicting, allowing them to be combined for better performance.

2) Adjustment strengths. EQL V1 overlooks discouraging gradients from background classes, and EQL V2 uses a function to calculate an adjusting weight to balance positive and negative gradient ratios, with its adjustment strength constrained by an upper limit of this function. Consequently, the adjustment strengths of both methods are relatively moderate. HP gains a more significant boost in positive gradient ratios for foreground and tail classes by eliminating discouraging gradients from background and head classes, as shown in Fig. 2 (a). Furthermore, as shown in Fig. 2 (b), EQL V1 and EQL V2 exhibit a relatively marginal improvement in the probability of foreground categories. In contrast, in HP, FIP surpasses both methods in enhancing the predicted probabilities of foreground categories. Moreover, IIP further increases the predicted probabilities of tail classes.

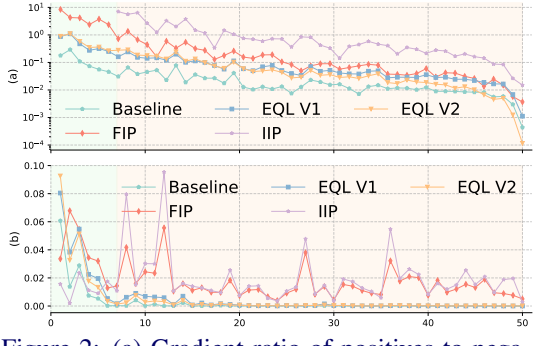

Figure 2: (a) Gradient ratio of positives to negatives for foreground classes. (b) The average predicted probability for foreground classes. The x-axis is the category index, arranged according to the instance count of each category.

3) Performance. HP outperforms both methods, as shown in Tab. 7. Specifically, HP gains 4.0 points higher on mR@K and 1.3 points higher on MR@K than EQL V2. Additionally, HP can be flexibly combined with the aforementioned methods to further enhance the learning of foreground and tail categories. For example, as shown in Tab. 7, the performance of mR@100 and MR@100 with HP+EQL V2 exceeds that of EQL V2 by 4.4 and 1.7 points, respectively.

6.3.2 Background-foreground Balancing Learning Methods.

FIP involves balancing learning between background and foreground classes. To prove the superiority of the FIP, we compare it with some possible alternative approaches.

Firstly, we compare FIP with Focal Loss [14], a classic method for addressing similar issues in object detection. As shown in Tab. 7, incorporating Focal Loss into CTP marginally improves mR@K but decreases R@K. Consequently, combining SGG models with Focal Loss does not effectively address the imbalance between background and foreground. The possible reason for such results is that it still relies on passive adjustment of the learning process through loss without actively tuning like our FIP.

Method	R@50/100	mR@50/100	MR@50/100
Baseline	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6
CTP(Focal Loss)	64.4 / 66.5	15.7 / 17.7	40.1 / 42.1
CTP($r = 0.25$)	65.6 / 67.4	17.2 / 18.6	41.4 / 43.0
CTP($r = 0.50$)	65.7 / 67.4	17.1 / 18.5	41.4 / 43.0
CTP($m = 2$)	65.6 / 67.3	17.2 / 18.6	41.4 / 43.0
CTP($m = 4$)	65.7 / 67.4	17.5 / 18.9	41.6 / 43.2
CTP (+FIP)	64.8 / 67.4	**18.7 / 21.3**	**41.8 / 44.4**
EQL V1 [23]	64.9 / 66.8	18.0 / 19.5	41.5 / 43.2
EQL V2 [23]	65.5 / 67.4	19.7 / 21.8	42.6 / 44.6
HP	64.2 / 66.0	24.1 / 25.8	44.2 / 45.9
HP + EQL V2	63.2 / 65.0	**28.1 / 30.2**	**45.7 / 47.6**

Table 7: Performance comparison of different basic prompts on the NPG task.

Secondly, we introduce background triplet sampling in each training batch. Specifically, we define the batch size of background triplets as B^- and the sampling ratio as r for the baseline model. We then randomly sample $B^- * r$ background triplets in each batch. As shown in Tab. 7, there are minimal changes observed in R@K and mR@K for CTP ($r = 0.25$) and CTP ($r = 0.50$). FIP surpasses the background sampling methods on R@K and mR@K.

Thirdly, we augment the training batch with copies of foreground triplets. As depicted in Tab. 7, we set the replication multiples m to either 2 or 4. It is observed that the R@K remains largely unchanged, while the mR@K and MR@K are inferior compared to FIP.

The second and third comparisons demonstrate that simply augmenting foreground triplets or removing background triplets does not effectively mitigate the background-foreground class imbalance issue in SGG. In contrast, our proposed FIP approach integrates additional positive learning gradients, enhancing the learning of foreground triplets and mitigating the detrimental impact of background triplets. This presents FIP as a more effective optimization-centric solution to the background-foreground unbalanced learning problem.

6.3.3 Multi-expert Model Methods.

There are certain similarities between our method and multi-expert models, such as GCL [5], and a model variant of our method (denoted as HFC), which replaces the prompts in HP as FC (fully connected layer). These models share a multi-expert model structure. Despite the structural similarity, our approach exhibits superior traits that set it apart from these methods.

Compared to GCL, HP offers three advantages: 1) GCL is specifically designed for SGG tasks, exhibiting overly strong task coupling and lacking the versatility to extend to other tasks. In contrast, our HP can be easily applied to other related tasks (e.g., the long-tailed classification discussed in Sec. 7. 2) GCL lacks novel predicate prediction capability, while

Models	PredCls			SGCls			SGDet		
	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100	R@50 / 100	mR@50 / 100	MR@50 / 100
Motifs [43]	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6	38.9 / 39.8	8.3 / 8.8	23.6 / 24.3	32.8 / 37.2	6.8 / 7.9	19.8 / 22.6
+HFC	62.1 / 64.3	23.2 / 25.4	42.7 / 44.9	38.3 / 39.3	13.3 / 14.2	25.8 / 26.8	32.4 / 36.8	7.9 / 9.3	20.2 / 23.1
+HP	64.2 / 66.0	**24.1 / 25.8**	**44.2 / 45.9**	39.3 / 40.2	**13.5 / 14.3**	**26.4 / 27.3**	31.9 / 36.3	**8.9 / 10.5**	**20.4 / 23.4**
+ GCL [8]	42.7 / 44.4	**36.1 / 38.2**	39.4 / 41.3	26.1 / 27.1	20.8 / 21.8	23.5 / 24.5	18.4 / 22.0	**16.8 / 19.3**	17.6 / 20.7
+ GCL + HP	50.4 / 52.5	34.5 / 37.0	**42.5 / 44.8**	26.8 / 27.8	**22.0 / 23.1**	**24.4 / 25.5**	21.3 / 24.9	16.3 / 18.8	**18.8 / 21.9**

Table 8: Comprehensive performance comparison of different types of multi-expert model methods in the informative SGG task on the VG dataset. The R@50/100, mR@50/100, and MR@50/100 on PredCls, SGCls, and SGDet tasks are reported.

Method	CIFAR-10-LT ↓	CIFAR-100-LT ↓
Baseline [8]	28.8	60.2
Rwt [8]	27.1	59.6
Focal Loss [24]	27.7	59.6
EQL V1 [29]	**26.9**	**58.7**
EQL V2 [27]	28.0	58.8
HP	23.0	56.1
HP+EQL V1	**21.9**	**55.4**

Table 9: Test set balanced error (averaged over 5 trials) on long-tailed CIFAR-10/-100 with ResNet-32 [8] as backbone.

Method	Head	Tail	All
Baseline [43]	25.05	23.96	24.17
Rwt [8]	23.81	22.84	23.02
Focal Loss [24]	22.40	21.81	21.93
EQL V1 [29]	21.97	23.03	22.84
EQL V2 [27]	**25.41**	**24.52**	**24.69**
HP	25.27	24.30	24.49
HP+EQL V2	**25.54**	**24.55**	**24.75**

Table 10: Comparison of different balancing learning methods in the object detection task on the VG dataset, the AP_{50} of different splits is reported.

our HP can predict novel predicates in a zero-shot inference manner. 3) The model design of GCL is excessively rigid, with high component coupling, impeding quick and seamless transferability. On the contrary, HP boasts flexibility and strong transferability. We can effortlessly integrate HP with GCL by substituting HC for GCL's classifier, as illustrated in Tab. 8, resulting in GCL+HP demonstrating significant performance improvements over GCL on R@K and MR@K.

Compared to HFC, HP offers two advantages: 1) HFC lacks the capability for novel predicate prediction, whereas HP possesses it. 2) HFC does not match our performance. As indicated in Tab. 8, HP surpasses HFC on all three metrics across the three tasks. The potential reason is that HP is a prompt-based learning method capable of leveraging informative textual modal knowledge obtained from large-scale language models. The integration of information from different modalities may be advantageous for informative SGG [32]. However, HFC does not incorporate such textual modal knowledge through prompt learning.

7 Generalization of HP to Other Tasks

Our method can be applied not only to the SGG task but also to other tasks, including:

7.1 Long-tailed Classification

Experimental Setting. We perform experiments on long-tailed image classification datasets, including CIFAR-10-LT [11] and CIFAR-100-LT [11], and the long-tailed data distribution of them is shown in Fig. 3 (b)-(c). We report the test set balanced error (averaged over 5 trials). The category with the sample count ranking in the top 20% is designated as the head classes for these two datasets. We use ResNet-32 [8] as the backbone model.

Experimental Results. We compare several commonly used methods for balanced classification in Tab. 9 and draw two observations: 1) Among these approaches, HP yields the best results on both long-tailed datasets. 2) Our method can be integrated with these approaches

to further enhance performance in long-tailed classification. For example, the combination of EQL V1+HP outperforms both HP and EQL V1 on both long-tailed datasets.

7.2 Long-tailed Detection

Experimental Setting. Furthermore, we apply our HP to the long-tailed detection task. Notably, the object classes in VG exhibit a long-tail distribution, as illustrated in Fig. 3 (a). Therefore, we conduct comparative experiments on the VG detection split to assess the performance of our HP method in comparison to several commonly employed approaches for addressing long-tailed detection. To better elucidate the performance improvements in tail-class detection, we report the AP_{50} for different class splits, including head classes (sample size $>$ 10K), tail classes (sample size $\leq$ 10K), and all classes.

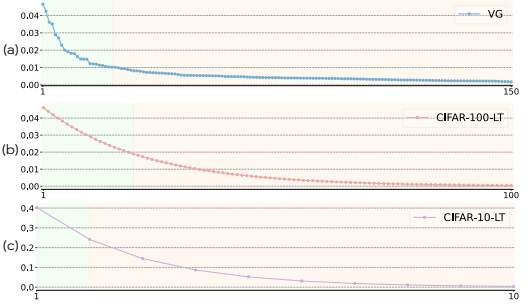

Figure 3: Distribution of different long-tailed datasets. The x-axis ticks are arranged in reverse order according to the number of samples. (a) The distribution of object instances in VG. (b) The distribution of classes in CIFAR-100-LT. (c). The distribution of classes in CIFAR-10-LT.

Experimental Results. As shown in Tab. 10, three key observations can be made: 1) Our HP significantly outperforms the baseline. 2) Our HP ranks second only to EQL V2 and surpasses all other methods in all three splits. 3) Our HP can be seamlessly integrated into other balanced learning methods for long-tailed detection. For instance, the combination of HP and EQL V2 yields additional improvements, achieving the best overall performance.

7.3 Zero-shot Relationship Retrieval

Experimental Setting. We employ the zero-shot recall, denoted as zs-R@K [25, 28], to evaluate the performance of all models on unseen triplets during training. Different from NPG, zero-shot relationship retrieval involves predicting unseen combinations of triplets with seen predicates [25, 28]. In this scenario, we compare performance with other methods addressing zero-shot relationship retrieval, including model-agnostic models TDE [25] and EMB [20], as well as model-specific models PeNet [30] and Dec [1].

Models	PredCls	SGCls	SGDet
	zs-R@50 / 100	zs-R@50 / 100	zs-R@50 / 100
Dec [1]	13.6 / 16.5	3.2 / 4.8	2.9 / 4.0
PeNet [30]	17.2 / 20.9	5.4 / 6.5	2.3 / 3.6
Motifs [28, 25]	10.9 / 14.5	2.2 / 3.0	0.1 / 0.2
Motifs (TDE) [25]	14.4 / 18.2	3.4 / 4.5	2.3 / 2.9
Motifs (EMB) [20]	4.9 / -	1.3 / -	0.2 / -
Motifs (HP-i)	18.9 / **22.0**	4.8 / 5.6	2.1 / 3.2
Motifs (HP)	**19.1** / 21.9	**4.9 / 5.9**	**2.2 / 3.5**
VCTree [28, 25]	10.8 / 14.3	1.9 / 2.6	0.2 / 0.7
VCTree (TDE) [25]	14.3 / 17.6	3.2 / 4.0	**2.6** / 3.2
VCTree (EMB) [20]	5.4 / -	1.9 / -	0.5 / -
VCTree (HP-i)	17.6 / 21.0	**7.5 / 8.7**	2.2 / **3.4**
VCTree (HP)	**18.9 / 22.0**	6.2 / 7.6	2.2 / 3.3

Table 11: Comparison of different methods under all three sub-tasks on the VG dataset. Model-specific models are marked in gray. Zero-shot Recall (zs-R@50/100) is reported.

Experimental Results. We derive three observations from Tab. 11: 1) Our HP significantly outperforms baseline. 2) HP performs better than other methods. On the Motifs baseline [28], HP outperforms the second-best model-agnostic method TDE by 15.3%, 31.1%, and 20.7% on PredCls, SGCls, and SGDet, respectively. HP also achieves comparable or superior performance with

Method	PredCls	SGCls	SGDet
	mR@50/100	mR@50/100	mR@50/100
KERN [8]	10.1 / 12.5	5.0 / 6.3	2.2 / 4.1
TDE [25]	11.5 / 13.4	5.3 / 6.8	2.7 / 3.9
Motifs [28]	9.6 / 11.9	4.5 / 5.8	1.8 / 3.3
Motifs (Dec) [2]	13.4 / 15.1	6.5 / 8.2	3.6 / 5.0
Motifs (**HP-i**)	**17.1 / 19.1**	**8.5 / 9.6**	**5.6 / 6.8**

Table 12: Comparison of various methods for relationship retrieval under the few-shot training data setting under all three sub-tasks.

Methods	PredCls		
	R@100	mR@100	MR@100
HP	64.2 / 66.0	**24.1** / 25.8	**44.2 / 45.9**
w / o BP	39.4 / 45.0	23.7 / **28.0**	31.6 / 36.5
w / o FIP	64.4 / 66.2	20.4 / 22.1	42.4 / 44.2
w / o IIP	64.8 / 67.4	18.7 / 21.3	41.8 / 44.4
w / o FIP + IIP	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6

Table 13: Ablation study of each component in HP.

the best model-specific model, PeNet. 3) The advantage of HP over HP-i on zero-shot retrieval is less apparent than in informative SGG, possibly due to the absence of tail group predicates in unseen triplets, resulting in diminished effectiveness of IIP in HP, thus bringing unchanged or decreased performance for HP.

7.4 Few-shot SGG Learning

Experimental Setting. Due to the limited training data in the tail group of SGG datasets, few-shot learning is a crucial and practical problem for the SGG task. To evaluate the effectiveness of HP in this scenario, we conduct experiments using few-shot training data. Specifically, we select $K^{'}$ images from each relation class to train the SGG models, with $K^{'}$ set to 10 in our study. It is important to note that in this task, the number of samples for each foreground category is equal. As a result, IIP is not utilized for this task, and only FIP is employed. Hence, we solely compare HP-i with other methods in this scenario.

Experimental Results. As depicted in Tab. 12, our approach outperforms the second-best method (Motifs (Dec)) by 4.0, 1.4, and 1.8 points across three tasks in terms of mR@100. This underscores the superior ability of HP to discriminate between different classes under the conditions of few-shot training data.

8 Ablation Studies for HP

8.1 Effect of Different Prompts in HP

Firstly, we conduct ablation experiments on each prompt in HP as presented in Tab. 13. We make two observations: 1) The effect of BP is significant. As evident in Row 1, the absence of BP leads to a substantial decrease in R@K, underscoring its efficacy in mitigating false negatives associated with background classes. 2) Both FIP and IIP are pivotal contributors to the enhancement of mR@K and MR@K. The removal of either one detrimentally affects overall performance. This directly validates the effectiveness of the FIP and IIP.

8.2 Different Pre-trained Language Models

We conduct experiments to evaluate the performance of different pre-trained language models in the HP-i. Specifically, we compare language models derived from CLIP [18] and BERT [3]. The results, as presented in Tab. 14, lead to the following observations: Firstly, compared to the baseline model, employing the prompt-based learning method with a language model derived from either CLIP or BERT improves mR@K and MR@K. This suggests that

Method	R@50/100	mR@50/100	MR@50/100
Baseline []	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6
BERT []	65.3 / 67.1	16.8 / 18.2	41.1 / 42.7
CLIP (RN-101) []	65.7 / 67.4	**17.2 / 18.4**	**41.5 / 42.9**
CLIP (RN-50) []	65.6 / 67.4	17.0 / **18.4**	41.3 / **42.9**

Table 14: Comparison of different pre-trained language models. The pre-trained language models are employed in conjunction with the CTP.

ω	Predcls		
	R@50/100	mR@50/100	MR@50/100
Baseline	65.2 / 67.0	14.8 / 16.1	40.0 / 41.6
0.05	**64.2 / 66.0**	24.1 / 25.8	**44.2 / 45.9**
0.10	63.3 / 65.3	24.1 / 26.1	43.7 / 45.7
0.15	63.4 / 65.4	**24.3 / 26.4**	43.9 / **45.9**

Table 15: Ablation study of different ω values.

the prompt-based learning method is more apt for the SGG task, which facilitates the training of robust SGG models capable of predicting more accurate scene graphs.

Secondly, we observe that the language model from CLIP is more effective. This may be attributed to the training process of CLIP, which involves both visual and textual input data, making it better suited for transferring to visual tasks. Thirdly, we conduct a comparison between the language models from CLIP associated with the visual models RN-101 and RN-50. We observe minimal accuracy differences between them and ultimately opt for RN-101 due to its higher MR@50.

8.3 Hyperparameter Selection for HP

In this section, we will use the HP with Motifs baseline on the VG dataset as an example to demonstrate the strategy for tuning the hyperparameters in HP. There are two sets of hyperparameters in HP. The first set is ω, which assigns weights to the various prompts in HP when calculating the overall loss function. As shown in Tab. 15, as ω increases, R@K decreases, while mR@K exhibits a slight increase. Ultimately, we choose $\omega = 0.05$, which corresponds to the highest MR@K (achieving the best results on both MR@50 and MR@100) and exhibits the least decline in R@K compared to the baseline.

α_1	SGCls		
	R@50/100	mR@50/100	MR@50/100
0.00	42.8 / 43.5	10.8 / 11.3	26.8 / 27.4
0.05	42.7 / 43.4	11.1 / 11.7	**26.9** / 27.6
0.10	42.5 / 43.4	**11.3** / 12.0	**26.9 / 27.7**
0.15	41.8 / 43.1	11.2 / **12.3**	26.5 / **27.7**
0.20	40.1 / 42.5	10.4 / 12.1	25.3 / 27.3
0.25	38.1 / 41.3	9.6 / 11.6	23.9 / 26.5

Table 16: Ablation study of different α_1 values in HP-i.

Next, we focus on α_1 and α_2 in ensemble inference. We tune these hyperparameters using the grid-search algorithm on the validation set on the SGCls task. Firstly, we adjust the hyperparameter α_1 for the HP-i method, as detailed in Tab. 16. It is observed that an increase in α_1 leads to an initial surge followed by a subsequent decline in mR@K, while R@K consistently decreases. Therefore, excessively increasing α_1 is not conducive to both mR@K and R@K. Hence, we select $\alpha_1 = 0.1$ as the optimal choice, achieving the best trade-off between R@K and mR@K and yielding optimal results on MR@K. Secondly, for the HP method, we tune both α_1 and α_2 parameters, as shown in Tab. 17. We observe that increasing α_1 leads to a

α_1	α_2	SGCls		
		R@50/100	mR@50/100	MR@50/100
0.00	0.00	42.4 / 43.1	11.4 / 12.0	26.9 / 27.6
0.05	0.00	42.3 / 43.0	11.8 / 12.3	27.1 / 27.7
0.10	0.00	42.3 / 43.0	12.4 / 13.0	27.4 / 28.0
0.15	0.00	42.2 / 42.9	12.8 / 13.4	27.5 / 28.2
0.20	0.00	42.1 / 42.8	13.2 / 13.8	**27.7 / 28.3**
0.25	0.00	41.9 / 42.6	**13.3 / 14.0**	27.6 / **28.3**
0.20	0.05	41.9 / 42.7	13.5 / 14.2	27.7 / 28.5
0.20	0.10	41.7 / 42.4	14.2 / 14.8	**28.0 / 28.6**
0.20	0.15	41.4 / 42.1	**14.5 / 15.1**	**28.0 / 28.6**

Table 17: Ablation study of different α_1 and α_2 values in HP.

decrease in R@K and an increase in mR@K and MR@K. We choose $\alpha_1 = 0.2$ as it attains the highest MR@K. Subsequently, while keeping $\alpha_1 = 0.2$ constant, we continue adjusting α_2. It is observed that an increase in α_2 results in a decrease in R@K and an increase in mR@K and MR@K. Ultimately, we choose $\alpha_2 = 0.1$, as it performs best on MR@K while minimizing the decrease in R@K.

Following a similar parameter adjustment strategy employed for α_1 and α_2 in the SGCls task, we extend the same approach to select parameters for both the PredCls and SGDet tasks. We choose the same parameters for the Predcls task as the SGCls task. In the SGDet task, we observe a heightened sensitivity to these two parameters. Ultimately, in the SGDet task, we choose $\alpha_1 = 0.0$ for IP and $\alpha_1 = 0.05$ and $\alpha_2 = 0.1$ for HP. When combined with Rwt methods [2], it will cause a substantial decrease in the performance of R@K metrics. Therefore, to prevent further decline in the R@K metric, we do not use the ensemble inference method in the Rwt methods (i.e., $\alpha_1 = 0.0$ and $\alpha_2 = 0.0$).

8.4 Analysis of Hyperparameters in HP-i and HP

In this subsection, we conduct a performance comparison between HP-i (w/o IIP) and HP (w/ IIP) across different α_1 parameter settings, as shown in Fig. 4. We derive three observations: 1) When comparing the baseline (i.e., $\alpha_1 = 0$ and $\alpha_2 = 0$), mR@K for both HP-i and HP surpasses those of the baseline. This validates the effectiveness of our proposed informative prompts in predicting informative predicates, encompassing FIP and IIP. 2) The reduction in R@K for HP-i is more pronounced than HP when increasing α_1. 3) For HP-i, as α_1 gradually increases, mR@K initially improves but then declines. In contrast, within the HP, an increase in α_1 exhibits a more stable and significant enhancement in both mR@K and MR@K. The explanation is that IIP further enhances the learning of tail classes compared to FIP by mitigating the negative impact of discouraging gradients from head classes, as discussed in Sec. 2, thereby further strengthening the performance of tail classes.

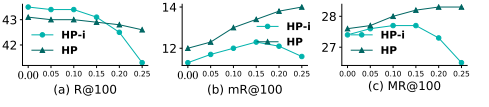

Figure 4: Comparison of HP-i and HP under different α_1, the R@100, mR@100, and MR@100 are reported. (SGCls task).

Methods	α_1	α_2	SGCls				
			R@100	mR@100	HR@100	TR@100	M
Baseline	-	-	39.8	8.8	39.6	3.4	22.9
HP-i	0.20	0.00	**40.6**	12.5	**40.4**	8.5	25.5
HP	0.20	0.00	**40.6**	13.0	**40.4**	9.1	25.8
HP	0.30	0.00	40.3	13.8	40.2	9.5	26.0
HP	0.20	0.10	40.2	**14.3**	39.6	**10.1**	**26.1**

Table 18: Comparison of baseline, HP-i, and HP under different α_1 and α_2 settings. HR@K and TR@K are the mean recall of head and tail classes, respectively. M represents the average of all metrics in the corresponding row.

Furthermore, to further validate the superiority of our proposed methods, we conduct a comprehensive comparison of baseline, HP-i, and HP performance under varied parameters on the test set. From Tab. 18, it is evident that: 1) Both HP-i and HP outperform the baseline on mR@100 and TR@100. This demonstrates the effectiveness of FIP and IIP. 2) Under the same parameter conditions (i.e., $\alpha_1 = 0.2$), HP exhibits superior performance (i.e., higher TR@100) for tail classes compared to HP-i; 3) The comparison of rows 4 and 5 reveals that setting $\alpha_2 = 0.1$ more effectively improves TR@100 than setting $\alpha_1 = 0.3$. The observations suggest that the inclusion of IIP enhances the learning of tail classes, corroborating conclusions similar to those presented in Fig. 5. Therefore, based on these observations, both FIP and IIP are indispensable in HP, jointly contributing to the learning of informative predicates.

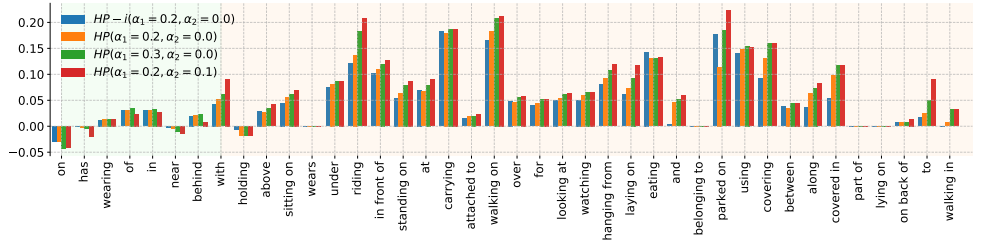

Figure 5: Absolute R@100 improvement compared with HP and HP-i on the VG dataset. The Top-45 relationship categories are selected according to their frequency of occurrence. The light green area is the head classes, and the light red area is the tail classes (SGCls task).

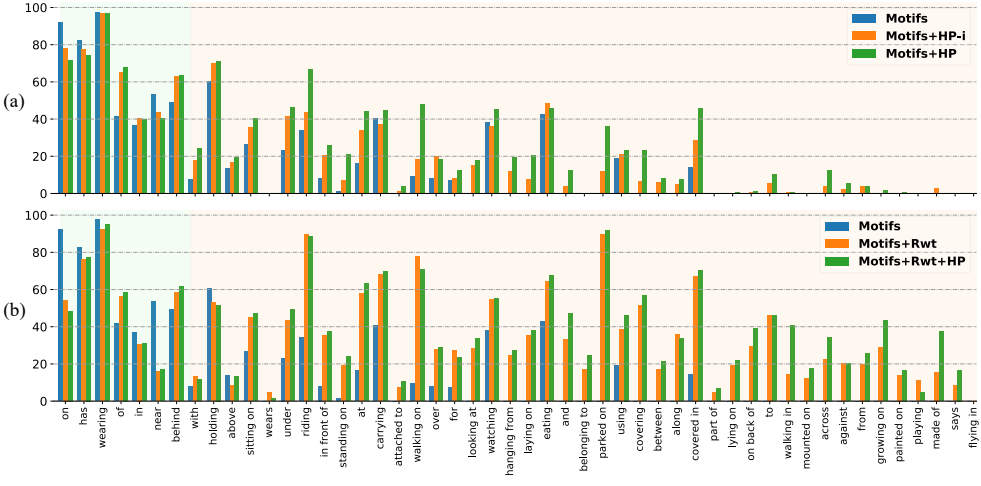

Figure 6: (a) Recall@100 of all predicate classes of Motifs, Motifs+HP-i, and Motifs+HP. (b) Recall@100 of all predicate classes of Motifs, Motifs+Rwt, and Motifs+Rwt+HP. Predicates are sorted in decreasing order of sample frequency (PredCls task).

8.5 Qualitative Results of Predicate Recall

For a more intuitive illustration of HP's infotmative predicate recognition capability, we provide Recall@100 for each predicate in comparison to baselines, including our proposed HP-i and HP, as depicted in Fig. 6 (a)-(b). From this statistical chart, we have several observations, as follows: 1) Our HP-i (only using FIP) outperforms Motifs on most foreground predicates, illustrating the effectiveness of our FIP in learning foreground classes. Intuitively, FIP provides enhanced positive reinforcement learning gradients for foreground classes. 2) With the incorporation of IIP, HP significantly outperforms HP-i on most tail predicates. This suggests that IIP further promotes the learning of tail classes by increasing the proportion of positive gradients for tail classes. 3) Upon integration with Rwt [7], HP's performance on most tail predicates experiences further improvement. This demonstrates the robust scalability and flexible transferability of our HP.

In conclusion, these results compellingly demonstrate that HP significantly enhances the learning of informative predicates, enabling the model to effectively distinguish informative predicates from uninformative ones.

8.6 A More Granular Informative Prompt

In this study, we propose a hierarchical prompt learning method that organizes prompts into three distinct levels of granularity. Each prompt is dedicated to specific, successively narrowed class groups, aiming to facilitate the learning of informative predicates. At the latter two levels, we employ prompts specifically tailored to enhance the learning of foreground and tail classes, respectively. To further investigate the impact of increased granularity, we introduce the fourth informative prompt, referred to as a More Granular Informative Prompt (MGIP). In this analysis, we subdivide the latter 50% of the tail group into a smaller subgroup and assign MGIP to train on these samples. Both training and prediction follow the same approach as with the FIP and IIP.

The experimental results in Tab. 19 demonstrate that the inclusion of the MGIP improves the mR@K. However, the MR@K shows a slight improvement (0.1 point on the MR@100) compared to the IIP. In contrast, IIP achieves more MR@K improvement (1.5 points on MR@100) compared to the FIP, indicating a more significant decline in the R@K of MGIP than IIP. These results also suggest that MGIP increases the performance loss in the head classes more than IIP. In conclusion, while a more granular level of informativeness enhances the performance of informative predicates, it also amplifies the magnitude of loss in the common predicates. We plan to delve deeper into this aspect in our future work.

Methods	PredCls		
	R@50/100	mR@50/100	MR@50/100
CTP	65.7 / 67.4	17.2 / 18.4	41.5 / 42.9
+FIP	64.8 / 67.4	18.7 / 21.3	41.8 / 44.4
+FIP+IIP	64.2 / 66.0	24.1 / 25.8	**44.2** / 45.9
+FIP+IIP+MGIP	62.5 / 64.5	**25.4 / 27.4**	44.0 / **46.0**

Table 19: Comparison of FIP, IIP, and MGIP on the PredCls task.

9 Further Clarification of HP

9.1 Characteristics of HP

To more comprehensively demonstrate the superiority of HP in tackling informative and novel predicate learning in SGG, we compare it with several alternative solutions, including model-agnostic prompts, a model-specific prompt SVRP [10] that possess the ability to transfer to novel classes, and FC (fully connected layer) that lacks the ability to transfer to novel classes. As illustrated in Tab. 20, HP exhibits all of the desirable characteristics, thereby establishing its superiority in informative and novel predicate learning tasks.

Desired Properties	FC	RP	DTP	CTP	SVRP	HP
Ability to predict novel predicates	✗	✓	✓	✓	✓	✓
Plug-and-play method	✓	✓	✓	✓	✗	✓
Learnable vector for stronger generalization	✓	✗	✗	✓	✓	✓
Computationally efficient and memory-efficient	✓	✓	✗	✓	✗	✓
Ability to predict more informative predicates	✗	✗	✗	✗	✗	✓
Establish a new SOTA for informative SGG	✗	✗	✗	✗	✗	✓

Table 20: Comparison of various learning models in SGG. Desired (undesired) properties are highlighted in green (red).

9.2 Detailed Algorithmic Procedure

For the convenience of readers' understanding, we present a detailed pseudocode for HP in Algorithm.2 [1].

[1] Due to space constraints, it appears in the last page.

Task	Split	Source	Predicates	Images	Prompts
NPG	Training	VG-50	$base_{50}$	57723	$\mathcal{T}_{base_{50}}$
	Test-base	VG-50	$base_{50}$	26646	$\mathcal{T}_{base_{50}}$
	Test-novel	VG-1800	$novel_{50}$	1465	$\mathcal{T}_{novel_{50}}$

Table 21: Dataset details for NPG task.

Task	Methods	PredCls		
		R@ 50 / 100	mR@ 50 / 100	MR@ 50 / 100
NPG (Novel)	HP (w/o freqbias)	13.3 / 13.4	7.9 / 7.9	10.6 / 10.6
	HP (w/ freqbias)	NA / NA	NA / NA	NA / NA
Infor-SGG	HP (w/o freqbias)	60.5 / 63.0	21.6 / 24.1	41.1 / 43.6
	HP (w/ freqbias)	**64.2 / 66.0**	**24.1 / 25.8**	**44.2 / 45.9**

Table 22: Influence of freqbias (frequency bias) on NPG and informative SGG tasks.

$base_{50}$	'above'	'across'	'against'	'along'	'and'	'at'	'attached to'
	'between'	'carrying'	'covered in'	'covering'	'eating'	'flying in'	'for'
	'hanging from'	'has'	'holding'	'in'	'in front of'	'laying on'	'looking at'
	'made of'	'mounted on'	'near'	'of'	'on'	'on back of'	'over'
	'part of'	'playing'	'riding'	'says'	'sitting on'	'standing on'	'to'
	'walking in'	'walking on'	'watching'	'wearing'	'wears'	'with'	'painted on'
	'growing on'	'belonging to'	'parked on'	'lying on'	'using'	'behind'	'from'
	'under'						
$novel_{50}$	'falling off'	'beneath'	'sitting on top'	'walking between'	'walking behind'	'at front of'	'closest to'
	'among'	'parked near'	'standing'	'cooking'	'sitting with'	'leaning against'	'walking towards'
	'standing behind'	'built into'	'held by'	'propped on'	'sitting next to'	'looking over'	'to left of'
	'mounted to'	'flying above'	'stacked on top of'	'leaning on'	'standing in front of'	'draped over'	'parked'
	'stopped at'	'operating'	'drinking from'	'standing in front'	'sitting'	'close to'	'sitting on top of'
	'hooked to'	'shining on'	'displayed on'	'giving'	'boarding'	'holding onto'	'right of'
	'turning'	'reaching for'	'touching'	'leaning up against'	'filled with'	'lifting'	'in center of'
	'looking down'						

Table 23: Different predicate groups appearing in the NPG task.

10 Further Clarification of NPG task

10.1 More Details of NPG

The training set of NPG is directly sourced from VG [6]. It contains 50 predicate classes and 150 object classes. The corresponding set of predicate categories for this dataset is denoted as $base_{50}$ as shown in Tab. 23. The novel evaluation set of NPG is derived from VG-1800 [29]. This newly introduced zero-shot predicate test set comprises 1465 images. We select the top 50 predicates based on frequency that are not included in the base set, and their predicate categories are denoted as $novel_{50}$ as presented in Tab. 23. Subsequently, employing $base_{50}$ and $novel_{50}$ as predicate words, we construct the base prompt $\mathcal{T}_{base_{50}}$ for training and the novel prompt $\mathcal{T}_{novel_{50}}$ for inference. For a more comprehensive summary, please refer to Tab. 21.

10.2 Additional Experiments

In this section, we investigate the effects of different components on the performance of NPG. These components include frequency bias [28] (denoted as freqbias), the ensemble inference method (denoted as ensemble), and the reweighting method (denoted as Rwt) [2].

Frequency bias. We observe that frequency bias is not transferable to NPG since novel predicates are not seen during training, and only the frequency of base predicates is accessible. However, when frequency statistics are available, as in formative SGG, which belongs to the closed-vocabulary setting, frequency bias contributes to performance gains on all three metrics, as demonstrated in Tab. 22. Therefore, we choose not to apply this method to the NPG task but to utilize it in the informative SGG task.

Ensemble inference. As shown in Tab. 24, no informative prompts for novel classes are available during the training stage of NPG, making ensemble inference inapplicable to NPG. Ensemble inference proves effective when the training dataset is relatively clean, such as in the informative SGG task as demonstrated in Tab. 24. Therefore, we opt not to employ this method in the NPG task but employ it in the informative SGG task.

Task	Methods	PredCls		
		R@ 50 / 100	mR@ 50 / 100	MR@ 50 / 100
NPG (novel)	HP (w/o ensemble)	13.4 / 13.4	7.9 / 7.9	10.6 / 10.6
	HP (w/ ensemble)	NA / NA	NA / NA	NA / NA
Infor-SGG	HP (w/o ensemble)	**65.7 / 67.4**	19.1 / 20.6	42.4 / 44.0
	HP (w/ ensemble)	64.2 / 66.0	**24.1 / 25.8**	**44.2 / 45.9**

Table 24: Influence of ensemble (ensemble inference) on NPG and informative SGG tasks.

Task	Methods	PredCls		
		R@ 50 / 100	mR@ 50 / 100	MR@ 50 / 100
NPG (Novel)	HP (w/o Rwt)	13.3 / 13.4	7.9 / 7.9	10.6 / 10.6
	HP (w/ Rwt)	**15.9 / 16.1**	**11.6 / 11.8**	**13.8 / 14.0**
Infor-SGG	HP (w/o Rwt)	**64.2 / 66.0**	24.1 / 25.8	44.2 / 45.9
	HP (w/ Rwt)	53.3 / 55.4	**37.3 / 39.3**	**45.3 / 47.4**

Table 25: Influence of Rwt on NPG and informative SGG tasks.

Debiasing methods. As shown in Tab. 25, debiasing methods improve all three metrics in NPG. The possible reason may be that Rwt makes an improvement in the performance of base tail classes, leading to better performance in NPG for novel categories semantically similar to these base tail classes. The application of debiasing methods in NPG is not the primary focus of this work; therefore, these strategies have not been utilized in both tasks in this work. However, based on the experimental results of Tab. 25, it appears that this direction is worth further exploration. We look forward to inspiring other researchers to explore this direction further.

11 Comparison Discussion of Basic Prompts

Based on the discussion in the preceding sections, we compare the traits of basic prompts in the SGG task, as illustrated in the Tab. 26. We summarize the following three points: 1) The high computational complexity of DTP hampers its application in the SGG task, while others with lower computational complexity are considered more prioritized. 2) The learnable nature of CTP enhances its learning ability, and class-specific CTP exhibits stronger learning ability than unified CTP, leading to better performance in informative SGG tasks. 3) When transferring to novel predicate learning tasks, the class-specific trait reduces the transferability of CTP, while both unified CTP and RP have high transferability. The difference between unified CTP and RP lies in their learnable characteristics, with the former having a higher dependency on data and the latter having a lower dependency. Therefore, RP performs better on the NPG task without novel training data.

Prompt	Complexity	Learning Ability	Transferability
RP	Low	Weak	Strong
DTP	High	Weak	Weak
CTP (class-specific)	Low	Strong	Medium
CTP (unified)	Low	Medium	Strong

Table 26: Comparison of characteristics for various basic prompts.

12 Qualitative Results

In Fig. 7 (a)-(c), we showcase examples predicted by the models Motifs [28], Motifs+HP. We have the following observations: Firstly, in the closed-vocabulary experimental setting, Motifs+HP exhibits the ability to predict informative relations (e.g., “standing on" and “with") rather than uninformative relations (e.g., “on" and “has"). This is evident in examples like “man-standing on-track" and “man-watching-man" in the left and right illustrations of Fig. 7 (a). Secondly, in the NPG experimental setting, Motifs+HP demonstrates the capability to predict unseen relations, e.g., “sitting next to" and “stacked on top of," which are not encountered during training. This is exemplified by instances like “woman sitting next to girl” and “basket stacked on top of elephant” in the left and right illustrations of Fig. 7 (b).

In summary, our proposed HP approach proves to be highly effective in generating scene

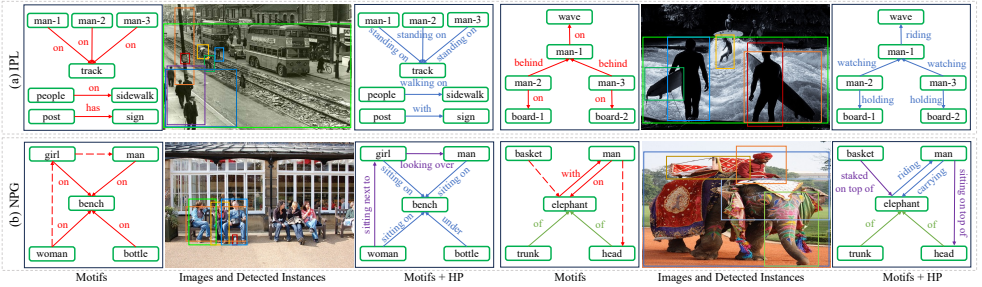

Figure 7: Qualitative comparisons between Motifs and our proposed Motifs+HP with regard to R@50 on the PredCls task. Correctly predicted uninformative relationships that match GT (i.e., Ground Truth) are depicted by solid green edges; wrongly predicted relationships that do not match GT are shown as solid red edges; and correctly predicted informative relationships that match GT are depicted by solid blue edges. We also note *logical* relationships that are not provided by GT yet are reasonable for prediction: *logical* novel relationships not predicted are highlighted by dashed red edges, *logical* novel relationships predicted are represented by solid purple edges. Due to space limitations, we exclude several detected objects that are less significant from the graphs.

graphs that are both distinguishable and specific. These graphs capture rich information and potentially reveal novel relationships, providing enhanced support for downstream tasks. These outstanding qualities of the HP method contribute to the generation of more accurate and contextually meaningful scene representations, underscoring its distinct advantages in advancing related downstream applications.

References

[1] Tianshui Chen, Weihao Yu, Riquan Chen, and Liang Lin. Knowledge-embedded routing network for scene graph generation. In *CVPR*, 2019.

[2] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277, 2019.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] Xingning Dong, Tian Gan, Xuemeng Song, Jianlong Wu, Yuan Cheng, and Liqiang Nie. Stacked hybrid-attention and group collaborative learning for unbiased scene graph generation. In *CVPR*, 2022.

[5] Xingning Dong, Tian Gan, Xuemeng Song, Jianlong Wu, Yuan Cheng, and Liqiang Nie. Stacked hybrid-attention and group collaborative learning for unbiased scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19427–19436, 2022.

[6] D.Xu, Y. Zhu, C. Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, 2017.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[9] Tao He, Lianli Gao, Jingkuan Song, Jianfei Cai, and Yuan-Fang Li. Semantic compositional learning for low-shot scene graph generation. In *ICCV*, 2021.

[10] Tao He, Lianli Gao, Jingkuan Song, and Yuan-Fang Li. Towards open-vocabulary scene graph generation with prompt-based finetuning. In *ECCV*, pages 56–73. Springer, 2022.

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

[12] R. Li, S. Zhang, B. Wan, and X. He. Bipartite graph network with adaptive message passing for unbiased scene graph generation. In *CVPR*, 2021.

[13] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. Oct 2017.

[15] Xin Lin, Changxing Ding, Jinquan Zeng, and Dacheng Tao. Gps-net: Graph property sensing network for scene graph generation. In *CVPR*, 2020.

[16] Xinyu Lyu, Lianli Gao, Yuyu Guo, Zhou Zhao, Hao Huang, Heng Tao Shen, and Jingkuan Song. Fine-grained predicates learning for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19467–19475, 2022.

[17] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

[18] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PMLR, 2021.

[19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[20] Mohammed Suhail, Abhay Mittal, Behjat Siddiquie, Chris Broaddus, Jayan Eledath, Gerard Medioni, and Leonid Sigal. Energy-based learning for scene graph generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13936–13945, 2021.

[21] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11662–11671, 2020.

[22] Jingru Tan, Xin Lu, Gang Zhang, Changqing Yin, and Quanquan Li. Equalization loss v2: A new gradient balance approach for long-tailed object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1685–1694, 2021.

[23] Jingru Tan, Bo Li, Xin Lu, Yongqiang Yao, Fengwei Yu, Tong He, and Wanli Ouyang. The equalization losses: Gradient-driven training for long-tailed object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[24] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. Learning to compose dynamic tree structures for visual contexts. In *CVPR*, 2019.

[25] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *CVPR*, 2020.

[26] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[27] Qifan Yu, Juncheng Li, Yu Wu, Siliang Tang, Wei Ji, and Yueting Zhuang. Visually-prompted language model for fine-grained scene graph generation in an open world. *arXiv preprint arXiv:2303.13233*, 2023.

[28] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *CVPR*, 2018.

[29] Ao Zhang, Yuan Yao, Qianyu Chen, Wei Ji, Zhiyuan Liu, Maosong Sun, and Tat-Seng Chua. Fine-grained scene graph generation with data transfer. In *ECCV*, 2022.

[30] Chaofan Zheng, Xinyu Lyu, Lianli Gao, Bo Dai, and Jingkuan Song. Prototype-based embedding network for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22783–22792, 2023.

[31] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.

[32] Xizhou Zhu, Jinguo Zhu, Hao Li, Xiaoshi Wu, Hongsheng Li, Xiaohua Wang, and Jifeng Dai. Uni-perceiver: Pre-training unified architecture for generic perception for zero-shot and few-shot tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16804–16815, 2022.

Algorithm 2 Pseudocode of HP in a PyTorch-like style.

```
# R: number of predicate classes; D: dimension of word embeddings;
# K: dimension of textual embeddings input to g; B_pos: number of foreground triplets
# B_tail: number of tail-class triplets; g: frozen pre-trained language model
# r: relation labels of input samples (N); r_t: word tokens of relation r (R)
# tau_bp, tau_fip, tau_iip: temperatures; omega: loss weight
# a1, a2: adjusting parameters in ensemble inference
# tri_t: word tokens of input triplets (N); DT: a flag indicating training phase or not
# f_vis: relation visual embeddings (N x K)

# word embeddings of learnable vector V_s: R x 1 x D
w_s = empty(R, 1, D)
# word embeddings of learnable vector V_o: R x 1 x D
w_o = empty(R, 1, D)
# word embeddings of relation text: R x 1 x D
w_r = TokenEmbedding(r_t)
# BP : R x 3 x D
bp = cat([w_s, w_r, w_o], dim=1)
# FIP : R x 3 x D
fip = cat([w_s, w_r, w_o], dim=1).deepcopy()
# IIP : R x 3 x D
iip = cat([w_s, w_r, w_o], dim=1).deepcopy()
# text embeddings of BP: N x K
t_bp= g(bp)
# text embeddings of FIP: N x K
t_fip = g(fip)
# text embeddings of IIP: N x K
t_iip = g(iip)

if DT # a flag representing training process
   # pick foreground visual embeddings: B_pos x K
   f_pos = GetPos(f_vis, r)
   # pick tail-class visual embeddings: B_tail x K
   f_tail = GetTail(f_vis, r)
   # pick foreground text embeddings: B_pos x K
   t_fip_pos = GetPos(t_fip, r)
   # pick tail-class text embeddings: B_tail x K
   t_iip_tail = GetTail(t_iip, r)

   # text embeddings of foreground triplet: B_pos x K
   t_pos = g(TokenEmbedding(GetPos(tri_t, r)))
   # text embeddings of tail-class triplet: B_tail x K
   t_tail = g(TokenEmbedding(GetTail(tri_t, r)))

   # normalized visual or text embeddings: B_pos x K
   t_pos = l2_normalize(t_pos, axis=1)
   f_pos = l2_normalize(f_pos, axis=1)
   f_tail = l2_normalize(f_tail, axis=1)
   t_fip_pos = l2_normalize(t_fip_pos, axis=1)
   t_iip_tail = l2_normalize(t_iip_tail, axis=1)

   # get the symmetric contrastive loss for FIP, ip_loss are defined in Algorithm 1
   loss_fip = ip_loss(t_pos, f_pos, t_fip_pos, tau_fip)
   # get the symmetric contrastive loss for IIP
   loss_iip = ip_loss(t_tail, f_tail, t_iip_tail, tau_iip)

   # cross-entropy loss of BP
   logits_bp = mm(f_vis, t_bp.T) * exp(tau_bp)
   loss_bp = CELoss(logits_bp, r, axis=0)

   # total loss
   loss = loss_bp + omega * (loss_fip + loss_iip)
else:
   # ensemble inference
   logits_bp = mm(f_vis, t_bp.T) * exp(tau_bp)
   logits_fip = mm(f_vis, t_fip.T) * exp(tau_fip)
   logits_iip = mm(f_vis, t_iip.T) * exp(tau_iip)
   # final logits
   logits = logits_bp.deepcopy()
   logits[:, 0] = logits_bp[:, 0]
   logits[:, 1:] = logits_bp[:, 1:] + a1 * logits_fip[:, 1:] + a2 * logits_iip[:, 1:]
```

`mm`: matrix multiplication; `cat`: concatenation; `empty`: returns an uninitialized tensor.; `TokenEmbedding`: returns the word embedding; `GetPos`: returns the foreground items; `GetTail`: returns the tail-class items; `arange`: returns a tensor of equally spaced values within a given range; `CELoss`: cross-entropy loss function.