Hierarchical Memory Learning for Fine-Grained Scene Graph Generation Supplementary Material

Youming Deng[1] Yansheng Li[1](✉) Yongjun Zhang[1] Xiang Xiang[2]
Jian Wang[3] Jingdong Chen[3] Jiayi Ma[4]

[1]School of Remote Sensing and Information Engineering, Wuhan University
[2]School of Artificial Intelligence and Automation, Huazhong University of Science and Technology [3]Ant Group
[4] Electronic Information School, Wuhan University

Abstract. The supplementary document is organized as follows: Sec. 1 gives a brief proof of approximate KL-Divergence; Sec. 2 explains the rationality and necessity of Mean@K; Sec. 3 verifies the fine-grained prediction ability by visualizing the qualitative results on each predicate; Sec. 4 compares the result of different CR Loss setting and their performance; Sec. 5 visualizes the qualitative result with different training stage numbers; Sec. 6 analyzes the variance of importance scores of parameters in the deep network; Sec. 7 provides a more comprehensive analysis about the training time.

1 Proof of Approximate KL-Divergence

Formally, the KL-Divergence [4] can be defined as:

$$D_{KL}(p_\theta(\mathbf{y}|\mathbf{x})||p_{\theta+\Delta\theta}(\mathbf{y}|\mathbf{x}) = \mathbb{E}_{(\mathbf{y},\mathbf{x})\sim D}\left[\log p_\theta\left(\mathbf{y}|\mathbf{x}\right) - \log p_{\theta+\Delta\theta}\left(\mathbf{y}|\mathbf{x}\right)\right]. \tag{1}$$

With the second Taylor Expansion of $\log p_{\theta+\Delta\theta}\left(\mathbf{y}|\mathbf{x}\right)$ at θ, we can get:

$$\log p_{\theta+\Delta\theta}\left(\mathbf{y}|\mathbf{x}\right) \approx \log p_\theta\left(\mathbf{y}|\mathbf{x}\right) + \Delta\theta^\top \frac{\partial \log p_\theta\left(\mathbf{y}|\mathbf{x}\right)}{\partial\theta} + \frac{1}{2}\Delta\theta^\top \frac{\partial^2 \log p_\theta\left(\mathbf{y}|\mathbf{x}\right)}{\partial\theta^2}\Delta\theta. \tag{2}$$

Then, substitute Eq. (2) into Eq. (1) and we get the approximate KL-Divergence:

$$\begin{aligned} D_{KL}(p_\theta(\mathbf{y}|\mathbf{x})||p_{\theta+\Delta\theta}(\mathbf{y}|\mathbf{x}) \approx{}& \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}[\log p_\theta\left(\mathbf{y}|\mathbf{x}\right)] - \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[\log p_{\theta+\Delta\theta}\left(\mathbf{y}|\mathbf{x}\right)\right] \\ &-\Delta\theta^\top \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[\frac{\partial \log p_\theta\left(\mathbf{y}|\mathbf{x}\right)}{\partial\theta}\right] - \frac{1}{2}\Delta\theta^\top \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[\frac{\partial^2 \log p_\theta\left(\mathbf{y}|\mathbf{x}\right)}{\partial\theta^2}\right]\Delta\theta. \end{aligned} \tag{3}$$

In Eq. (3), the first order partial derivative can be eliminated by:

$$\begin{aligned}\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[\frac{\partial \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta}\right] =& \mathbb{E}_{\mathbf{x}\sim D}\left[\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})\frac{\partial \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta}\right] \\ =& \mathbb{E}_{\mathbf{x}\sim D}\left[\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})\frac{1}{p_\theta(\mathbf{y}|\mathbf{x})}\frac{\partial p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta}\right] \\ =& \mathbb{E}_{\mathbf{x}\sim D}\left[\frac{1}{\partial\theta}\sum_{\mathbf{y}} \partial p_\theta(\mathbf{y}|\mathbf{x})\right] \\ =& \mathbb{E}_{\mathbf{x}\sim D}[0] = 0.\end{aligned} \tag{4}$$

In addition, the second order partial derivative in Eq. (3) can be replaced by Fisher Matrix F_θ:

$$\begin{aligned}\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[-\frac{\partial^2 \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta^2}\right] = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[-\frac{1}{p_\theta(\mathbf{y}|\mathbf{x})}\frac{\partial^2 p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta^2}\right] \\ +\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[\left(\frac{\partial \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta}\right)\left(\frac{\partial \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta}\right)^\top\right] = 0 + F_\theta.\end{aligned} \tag{5}$$

By substituting Eq. (4) and Eq. (5), we get the approximate KL-Divergence:

$$\begin{aligned}D_{KL}\left(p_\theta(\mathbf{y}|\mathbf{x})||p_{\theta+\Delta\theta}(\mathbf{y}|\mathbf{x})\right) &\approx \frac{1}{2}\Delta\theta^\top \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D}\left[-\frac{\partial^2 \log p_\theta(\mathbf{y}|\mathbf{x})}{\partial\theta^2}\right]\Delta\theta \\ &= \frac{1}{2}\Delta\theta^\top F_\theta \Delta\theta.\end{aligned} \tag{6}$$

2 Rationality and Necessity of Metric Mean@K

People tend to annotate the relationships with high-frequent predicates with an identical visual relationship in an image instead of more informative ones [5]. For example, "dog-on-bench" is more likely to be annotated than "dog-sitting on-bench". Because of this labeling phenomenon, a model with a fine-grained prediction preference will perform poorly on the "head part" (e.g., "on" and "has"). The reasonable and fine-grained predictions will be regarded as mistakes, which leads to the reporting bias of Recall@K (R@K). Thus, taking R@K as the primary evaluation metric is not plausible. For a better evaluation, [7] adopted mean Recall@K (mR@K) as the primary metric for the first time.

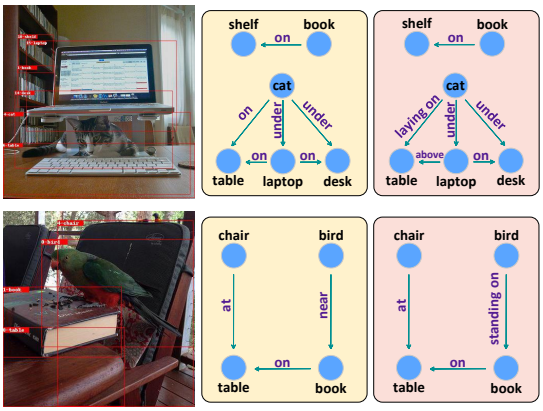

Fig. 1. Image with both position and verb as different granularity. Yellow scene graph is predicted by VCTree [6], while pink one is predicted by VCTree-HML.

Unfortunately, there is an intriguing fact that lies in our semantic expressions. Some objects with verb property typically have more fine-grained labeling, while other objects typically have position labeling. We can find several specific occasions like Fig. 1. The relationships between daily life objects are mostly positional, such as "book-on-shelf" or "laptop-on-desk". In this circumstance, "standing on" or "growing on" is unsuitable. In contrast, animals like "bird" and "cat" or humans, including "kid" and "man" have more verb relationships such as "sitting on" and "laying on". Hence, considering mR@K alone is also not very comprehensive.

Since mR@K and R@K restrict mutually, any methods with higher mR@K will get less R@K, and vice versa. With the consideration above, we adopt Mean@K, which calculates the average score of mR@K and R@K under an identical K to evaluate our HML further and provide a fair comparison.

3 Verification of Fine-Grained Prediction Ability

For better analysis and comparison, we compare the improvement of R@100 [8] on each predicate, respectively. The experiments were carried out on three models that are used in body part of our paper: Motif [12], Transformer [9,2], and VCTree [6].

In Figs. 2 to 4, we visualize the qualitative results of traditional training model and model with our HML on each predicate. We do not use the name of predicates for simplicity. Instead, we rank all predicates in the order of frequency, and the X-axis number represents each predicate's ranking. It is worth noticing that the performance improvements are impressive. Some "tail part" predicates that can not be correctly recognized in the original training framework drastically rise. Besides, the reason for the decline in the "head part" including "on" and "has" has been discussed in the body part of our paper.

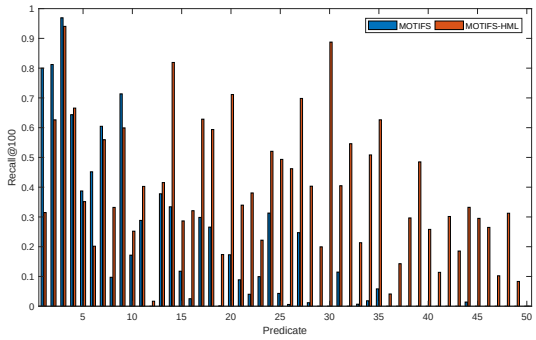

Fig. 2. Motif: Recall@100 [8] on Predicate Classification for all 50 predicates.

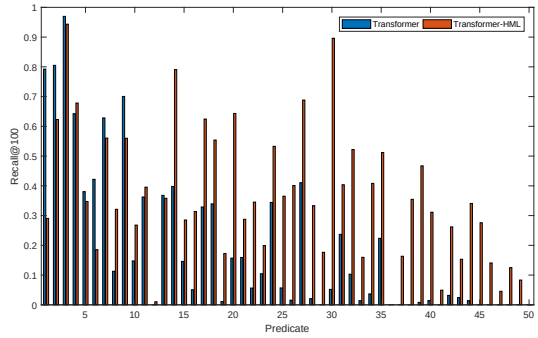

Fig. 3. Transformer: Recall@100 [8] on Predicate Classification for all 50 predicates.

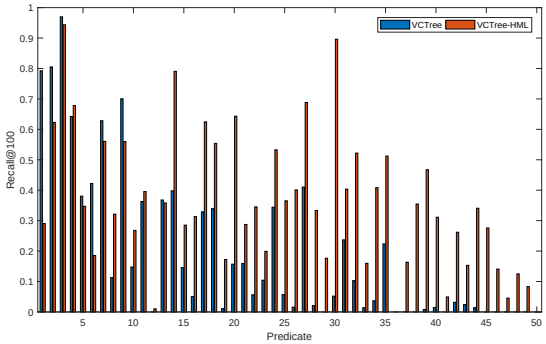

Fig. 4. VCTree: Recall@100 [8] on Predicate Classification for all 50 predicates.

4 Ablation on CR Loss

In order to find a suitable form of CR Loss, we conduct several experiments on other possible CR Loss. As is shown in Tab. 1, even though Cosine Loss and L1 perform better on Recall, L2 Loss can achieve more excellent trade-off on both metrics. The result verified our hypothesis in the main body.

CR Loss	Predicate Classification						Mean@20	Mean@50	Mean@100
	mR@20	mR@50	mR@100	R@20	R@50	R@100			
Cosine	25.3	30.5	32.7	**41.3**	**49.3**	**51.9**	33.3	39.9	42.3
KL	25.8	32.3	34.7	35.0	43.8	46.5	30.4	38.1	40.6
L1	28.9	34.7	37.1	39.9	46.4	48.4	34.4	40.6	42.8
Smooth L1	29.4	36.0	38.3	38.2	45.1	47.1	33.9	40.6	42.7
L2 (The recommended one)	**30.1**	**36.3**	**38.7**	40.5	47.1	49.1	**35.3**	**41.7**	**43.9**

Table 1. Ablation Study of CR under MOTIFS-HML

5 Qualitative Result of Three Training Stage

The ablation of stage number indicates the suitable stage number should be 2 in VG dataset. We visualize some of the qualitative results of different stages in Fig. 5. Compared with one stage of training, other stages can predict fine results, but too many stages jeopardize the hierarchical training and partly lead to degradation.

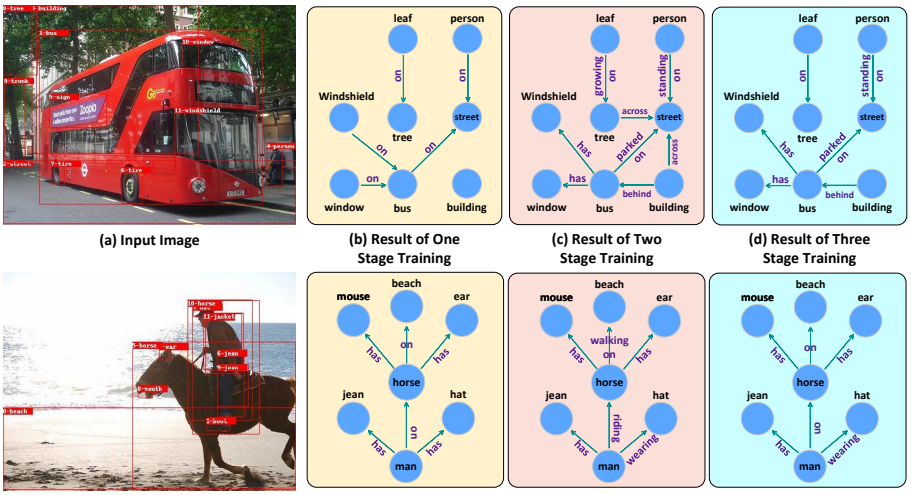

Fig. 5. Comparison of Qualitative on Different Stage Number.

6 Variance of Importance Scores of Parameters

As illustrated in the Model Reconstruction Loss section, an importance score is an efficient way to evaluate each parameter's importance in a model. To verify our assumption that the parameters in the target for different predicate classes, we visualize Figs. 6 to 8. For simplicity, we omit backbone layers and only pick up the weights concerning relation prediction. We first calculate the fractions of each parameter in the first and second stages:

$$Frac(p_n) = \frac{p_n^1}{p_n^2}, \tag{7}$$

where n represents n^{th} one of all parameters and superscripts 1 and 2 represent the stage number. Then find the mean of $Frac(p_n)$ of each layer (the number of the X-axis in the figure represents the layer number). The result of Figs. 6 to 8, shows that the importance scores of different parameters in each layer fluctuate a lot, verifying the assumption that different models parameters specialize in different tasks (i.e., different predicates).

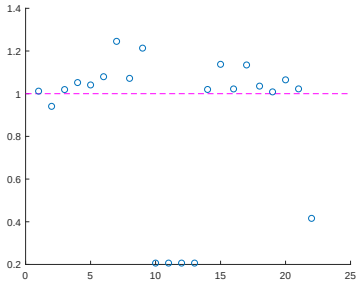

Fig. 6. The mean of importance scores in each relation prediction layer of MOTIFS [12].

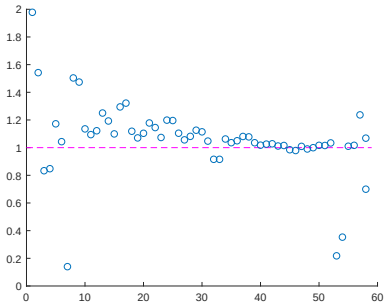

Fig. 7. The mean of importance scores in each relation prediction layer of Transformer [9,2].

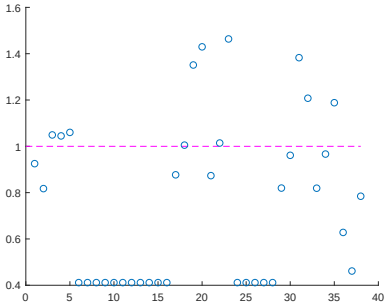

Fig. 8. The mean of importance scores in each relation prediction layer of VCTree [6].

7 Time Analysis

Intuitively, our HML tends to be extremely time-consuming due to hierarchical learning. In order to explore the computational complexity of our HML, we compare all competitive methods that have publicly released the source code in the last two years. We train all the methods under the same GPU computing environment and record the running times. We further analyze the training time per image on MOITFS with different frameworks for a more comprehensive study. Because of the different training conditions, it is hard to compare all methods fairly. For instance, some methods can be trained quickly for every image, but converging takes longer.

As shown in Tab. 2, our HML spends more than twice the training time on one image than other methods. However, the property of the HML framework is worth noticing. In the HML, a model is trained under the guidance of a well-trained model and within a relatively balanced and small fraction of data. Therefore, it takes fewer iterations for a model under the HML to converge, making the training increase insignificant.

All the experiments were carried out on identical NVIDIA Tesla V100 GPUs to pursue a fair comparison. Due to the limited memory of our GPU, we can not fully carry out identical experiment settings in [10] set batch-size to be 12 instead of the original 48 setting.

Model+Framework	mR@50/100	Time/img (s)	Overall Time (hr)
MOTIFS [12]	15.9/17.2	0.101	8.636
MOTIFS-TDE [7]	25.5/29.1	0.105	9.542
MOTIFS-CogTree [11]	26.4/29.0	0.124	8.487
MOTIFS-DLFE [1]	26.9/28.8	0.103	7.731
MOTIFS-BPL-SA [3]	29.7/31.7	0.211	11.135
PCPL [10]	35.2/37.8	0.119	8.163
MOTIFS-HML (Ours)	36.9/39.2	0.223	10.933

Table 2. Time evaluation on Predicate Classification. We compare different methods with our HML of training time per image and overall time.

References

1. Chiou, M.J., Ding, H., Yan, H., Wang, C., Zimmermann, R., Feng, J.: Recovering the unbiased scene graphs from the biased ones. In: ACMMM. pp. 1581–1590 (2021) 7
2. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: ICLR (2021) 3, 6
3. Guo, Y., Gao, L., Wang, X., Hu, Y., Xu, X., Lu, X., Shen, H.T., Song, J.: From general to specific: Informative scene graph generation via balance adjustment. In: ICCV. pp. 16383–16392 (2021) 7
4. Kullback, S., Leibler, R.A.: On information and sufficiency. The annals of mathematical statistics **22**(1), 79–86 (1951) 1
5. Misra, I., Lawrence Zitnick, C., Mitchell, M., Girshick, R.: Seeing through the human reporting bias: Visual classifiers from noisy human-centric labels. In: CVPR. pp. 2930–2939 (2016) 2
6. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: CVPR (July 2017) 3, 7
7. Tang, K., Niu, Y., Huang, J., Shi, J., 8 Zhang, H.: Unbiased scene graph generation from biased training. In: CVPR. pp. 3716–3725 (2020) 2, 7
8. Tang, K., Zhang, H., Wu, B., Luo, W., Liu, W.: Learning to compose dynamic tree structures for visual contexts. In: CVPR. pp. 6619–6628 (2019) 3, 4
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NIPS. pp. 5998–6008 (2017) 3, 6
10. Yan, S., Shen, C., Jin, Z., Huang, J., Jiang, R., Chen, Y., Hua, X.S.: Pcpl: Predicate-correlation perception learning for unbiased scene graph generation. In: ACMMM. pp. 265–273 (2020) 7
11. Yu, J., Chai, Y., Wang, Y., Hu, Y., Wu, Q.: Cogtree: Cognition tree loss for unbiased scene graph generation. In: IJCAI. pp. 1274–1280 (2020) 7
12. Zellers, R., Yatskar, M., Thomson, S., Choi, Y.: Neural motifs: Scene graph parsing with global context. In: CVPR. pp. 5831–5840 (2018) 3, 6, 7