

Appendix: Calibration for Long-tailed Scene Graph Generation

For a better understanding of the main paper, we provide additional details in this appendix, including:

- Sec.1 provides **further implementation details**.
- Sec.2 provides **additional analysis of COC components**.
- Sec.3 provides **additional experiments on calibration**.
- Sec.4 provides **additional analysis on balanced learning**.
- Sec.5 provides **hyperparameter selection**.
- Sec.6 provides **qualitative results**.

1 FURTHER IMPLEMENTATION DETAILS

We employ a pretrained Faster-RCNN [10] with ResNeXt-101-FPN [5] as the backbone, whose parameters are fixed during training as in previous works [4, 11, 14]. Our models are trained using SGD optimization with a maximum of 24K (in VG) and 30K (in GQA-200) iterations. The batch size for sampled images is set to 16, while the batch size for sampled triplets per image is 512. The initial learning rate is set to 0.001 and decays by a factor of 10 at the 10Kth and 16Kth iterations. All experiments are conducted using V100 GPUs. β is set to 0.99999 for both PredCls and SGCl tasks, and 0.999999 for the SGG task. \mathcal{M} is set to 15.0 and α is set to 0.75 (in VG) or 0.60 (in GQA-200) in the piecewise curriculum function. Further details on hyperparameter selection can be found in Sec. 5.

We obtain results from two prominent SGG datasets, which are Visual Genome (VG) [3] and GQA-200 [2]. VG consists of 57723 images for training, 5000 images for validation, and 26646 images for testing. It contains 150 object categories and 50 relation classes. GQA-200 consists of 57623 training images, 5000 validation images, and 8208 test images. It contains 200 object categories and 100 relation classes. We separate the categories into two separate groups based on the number of instances in the training split: head classes (more than 10K samples) and tail classes (less than 10K samples).

2 ADDITIONAL ANALYSIS OF COMPONENTS

In the main paper, we present comprehensive qualitative and quantitative results to validate the effectiveness of the proposed methods. In this section, we will provide additional analysis to further substantiate the efficacy of each component within COC.

2.1 Additional analysis of PC

Tab. 1 displays the performance of each component in PC, including training on the hypersphere and employing discriminative regularization. Both components contribute to improving SGECE and enhancing mR@K and MR@K, underscoring their effectiveness in enhancing model calibration and addressing the issue of unbalanced learning for long-tailed SGG. Furthermore, compared to the baseline method, the inclusion of the PC module promotes the learning of more discriminative features and establishes a representation space characterized by tighter boundaries and reduced overlap between classes, as depicted in Fig. 1 (i).

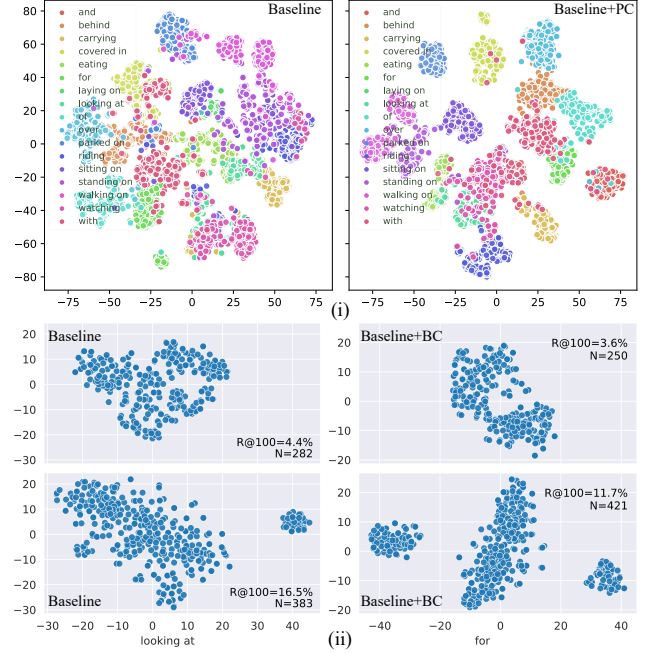


Figure 1: i. The visualization displays clustered features corresponding to different classes in VG. Compared to the Motifs [13] baseline model, the proposed PC method more effectively disperses features between different classes and clusters features within the same class. **ii.** The visualization displays recalled samples (the number is N) of the ‘looking at’ and ‘for’ classes within the tail group. Compared to the baseline (Motifs+PC), the proposed BC method effectively boosts the recall of tail-class samples, resulting in improvements in R@100 (i.e., Recall@100) for these tail classes.

Models	PredCls			
	R@50/100	mR@50/100	MR@50/100	SGECE
Baseline	65.2 / 67.2	14.9 / 16.3	40.1 / 41.8	75.6
+Hypersphere	65.5 / 67.3	19.2 / 20.6	42.4 / 44.0	50.8
+Discriminative Regular	65.4 / 67.3	20.4 / 22.0	42.9 / 44.7	49.5

Table 1: The ablation study of components in PC module. The baseline model is Motifs [13].

2.2 Additional analysis of BC

As shown in Fig. 1 (ii), the proposed BC module can increase the number of recalled tail class samples. This indicates that the BC module, designed to create separation among negative samples and indirectly decrease the feature deviation of tail classes, effectively reverts a greater number of misclassified tail samples back within their decision boundaries. Moreover, as shown in Fig. 2, the addition of the BC module enhances the predicted confidence levels for tail classes and improves both SGECE and tR@K 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 [13]	66.1 / 68.0	14.6 / 15.8	40.4 / 41.9	39.3 / 40.1	8.0 / 8.5	23.7 / 24.3	32.3 / 37.3	5.8 / 7.1	19.1 / 22.2
+Rwt [1, 15]	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
+COC(w/o ϵ)	58.0 / 60.0	36.2 / 38.3	47.1 / 49.2	35.2 / 36.0	17.8 / 20.5	26.5 / 28.3	27.2 / 31.3	14.7 / 17.1	21.0 / 24.2
+COC(w/ ϵ)	59.7 / 62.0	35.3 / 38.3	47.5 / 50.2	36.8 / 37.8	19.6 / 20.5	28.2 / 29.2	27.6 / 32.0	15.1 / 17.4	21.4 / 24.7

Table 2: Performance comparison of different methods on the VG dataset. Rwt denotes the reweighting method [1].

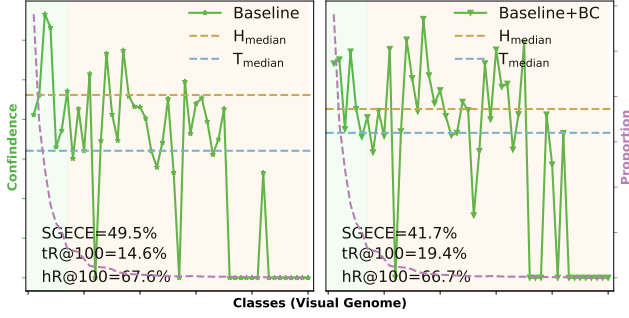


Figure 2: Predicted confidence distribution of head (in light green span) and tail (in light red span) classes. Confidences are represented by median confidences in each class. The baseline model is Motifs+PC.

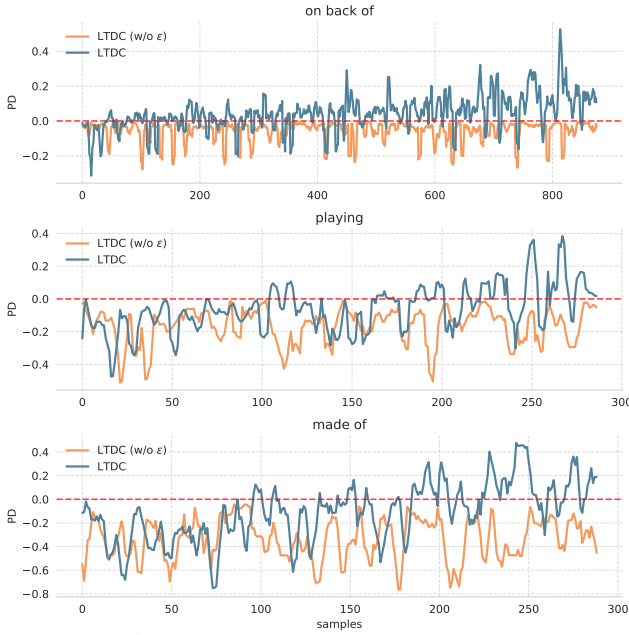


Figure 3: PD (Probability Discrepancy) of different tail classes during model training.

2.3 Additional analysis of TDC

For the TDC module, we will provide additional analysis in this section, including three aspects:

1) Effectiveness of transferring factor ϵ . From Fig. 5 (i), it is evident that in the baseline model, the tail classes are predominantly predicted as head classes (as indicated by the majority of predictions in tail classes being marked in orange). This observation validates the motivation behind the TDC module: current models overlook the distorted target distribution issue and rely solely on relation distribution for training, which leads trained models to be overconfident for head classes and biased towards predicting them.

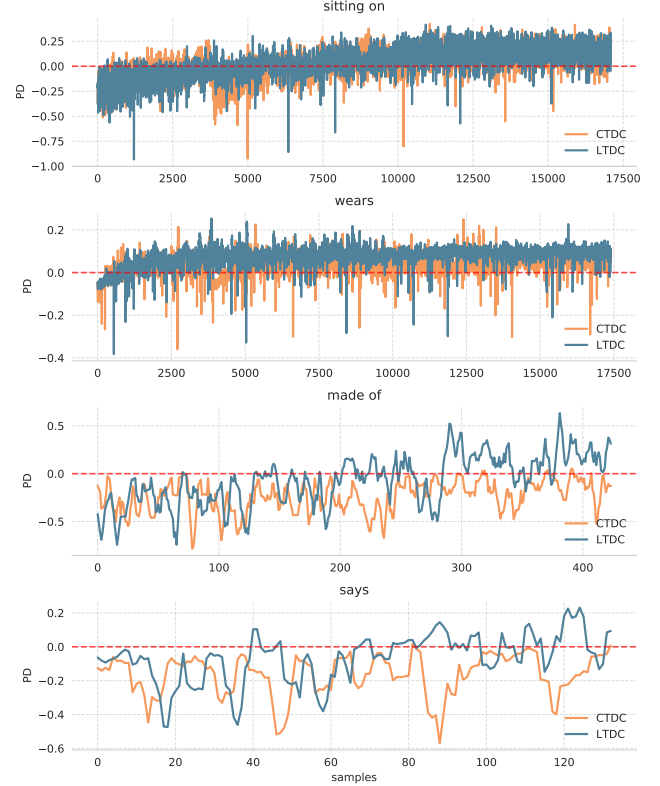


Figure 4: PD (Probability Discrepancy) of different tail classes during model training. The ‘sitting on’ and ‘wears’ are tail classes with a larger number of samples, while the ‘made of’ and ‘says’ are tail classes with a smaller number of samples.

Then, we compare the tail class performance between the LTDC (w/o ϵ) and LTDC (w/ ϵ) in Fig. 5 (ii)-(iii). It can be seen that LTDC (w/ ϵ) reduces the samples where tail classes are incorrectly predicted as head classes, further improving recall of tail classes. To gain a more intuitive understanding of how factor ϵ functions during the optimization process, we define a metric called **PD (Probability Discrepancy)** in Eq. 1, which can quantitatively assess how well the current model fits the transferred targets during training.

$$PD(\mathbf{p}, \mathbf{q}) = \mathbf{p} - \mathbf{q}, \quad (1)$$

where \mathbf{p} is the predicted probability and \mathbf{q} is the true probability derived from the within-triplet prior. A PD less than 0 indicates poor fitting, while a larger PD indicates better fitting. We visualize the PD for some tail classes in Fig. 3. It can be observed that without using the factor ϵ , the PD is subpar (i.e., $PD < 0$). However, after using the factor ϵ , the PD is enhanced (i.e., $PD > 0$). Therefore, the designed ϵ factor can help the model better fit the transferred target distribution. Moreover, the SGECE is also enhanced, indicating that transferring the distorted targets helps improve calibration.

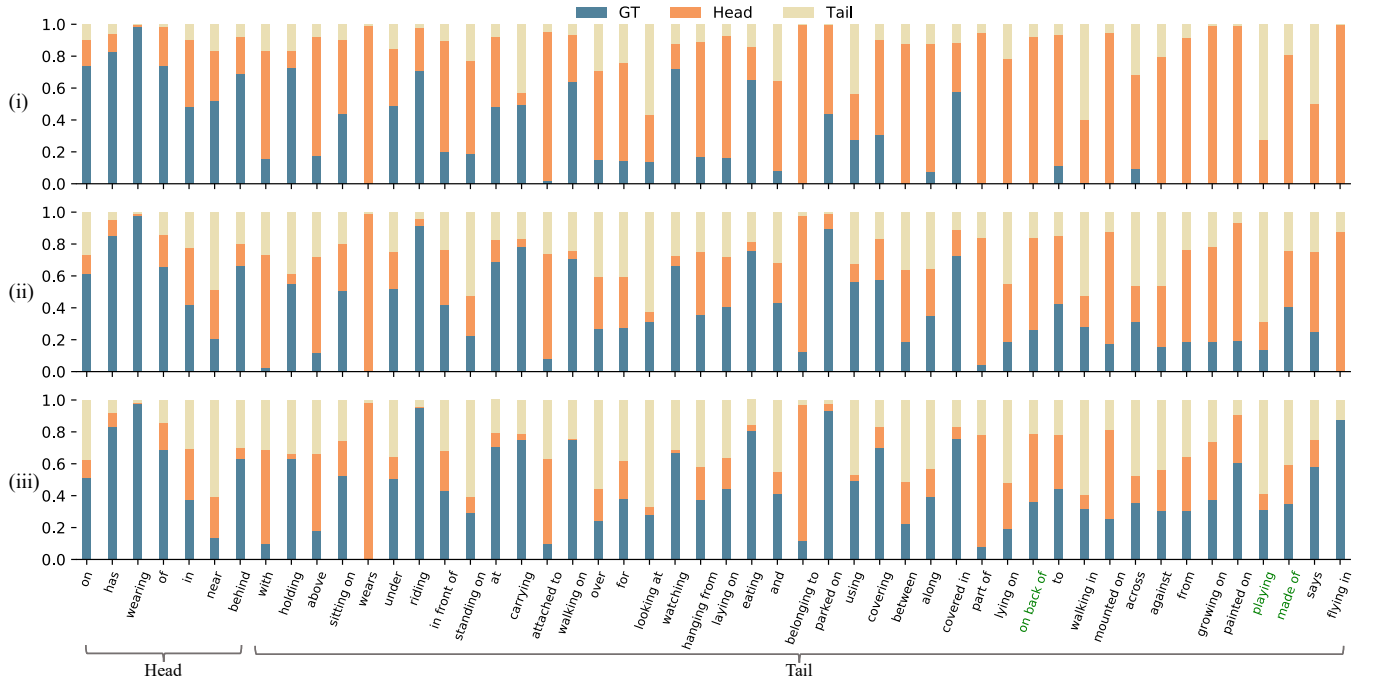


Figure 5: The distribution of predicted classes for each class. Blue marks indicate the proportion of predicted labels that match the Ground Truth (GT) labels. Orange marks indicate the proportion of predicted labels that belong to head classes and do not match the GT. Yellow marks indicate the proportion of predicted labels that belong to tail classes and do not match the GT. (i) Predictions derived from Baseline (Motifs+PC+BC). (ii) Predictions derived from Baseline+LTDC (w/o ϵ). (iii) Predictions derived from Baseline+LTDC (w/ ϵ). The PD performances of green-marked classes are illustrated in Fig. 3.

Models	PredCls					
	R@100	mR@100	MR@100	hR@100	tR@100	SGECE
Baseline	66.1	26.0	46.1	66.7	19.4	41.7
+CTDC	65.6	28.2	46.9	63.1	22.5	45.9
+LTDC	60.2	34.5	47.4	54.4	31.2	31.8

Table 3: Performance comparison of CTDC and LTDC. The baseline model is Motifs+PC+BC.

Models	PredCls					
	R@100	mR@100	MR@100	hR@100	tR@100	SGECE
Baseline	66.1	26.0	46.1	66.7	19.4	41.7
+LTDC (w/o CTR)	60.2	34.5	47.4	54.4	31.2	31.8
+LTDC (w/ CTR)	62.0	38.3	50.2	58.9	35.0	26.6

Table 4: Influence of CTR in the TDC module. The baseline model is Motifs+PC+BC.

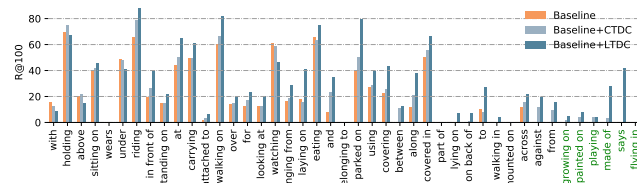


Figure 6: The R@100 performance of each tail class. The baseline model is Motifs+PC+BC. The green-marked classes denote sparse tail classes (with fewer than 200 samples).

In addition, we comprehensively compare the performance of the COC method on balanced learning with or without using the transferring factor ϵ in Tab. 2. It can be observed that using the factor ϵ

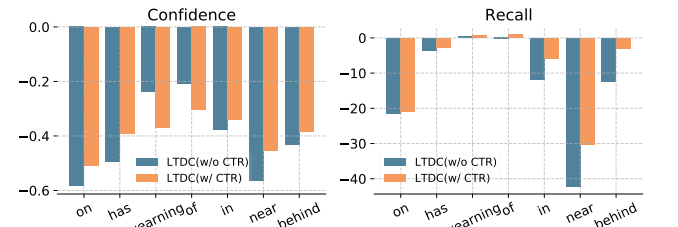


Figure 7: The change in predicted confidence and recall (R@100) relative to the baseline model (Motifs+PC+BC). Confidences are represented by median confidences in each class. This demonstrates the effectiveness of ϵ in TDC on balanced learning. Finally, we also compare COC with the Rwt [1] method (c.f. Tab. 2) and observe that Rwt performs less effectively than COC. The specific reason is that Rwt sets loss weights only based on relation distribution, which does not solve the issue of distorted target distribution. Instead, COC can adaptively modify the training targets during training to align with the targeted within-triplet distribution.

2) Comparison of CTDC and LTDC. From Tab. 3, we observe a gap in tR@K and MR@K between the CTDC and LTDC modules. This difference arises because the CTDC module overlooks distribution variations among tail classes, resulting in insufficient learning for the transferred target of harder-to-learn tail classes. Therefore, CTDC may not effectively solve underconfidence in tail classes and model miscalibration. We plot the PD of different tail classes in Fig. 4 and have two observations: 1) For abundant tail classes, both CTDC and LTDC fit the transferred targets nicely. 2)

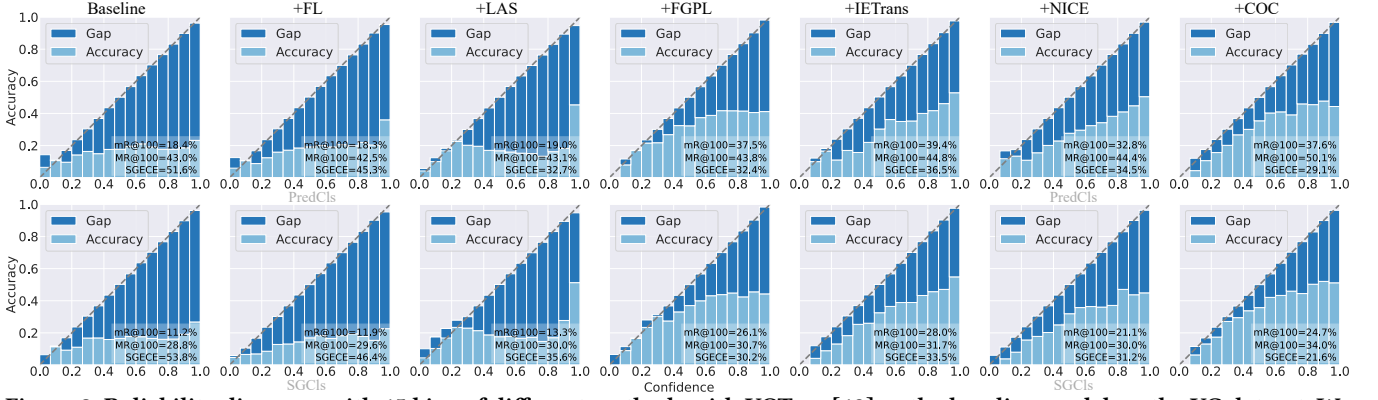


Figure 8: Reliability diagrams with 15 bins of different methods with VCTree [12] as the baseline model on the VG dataset. We reproduce other works on our platform based on their open-source projects to obtain new SGECE results.

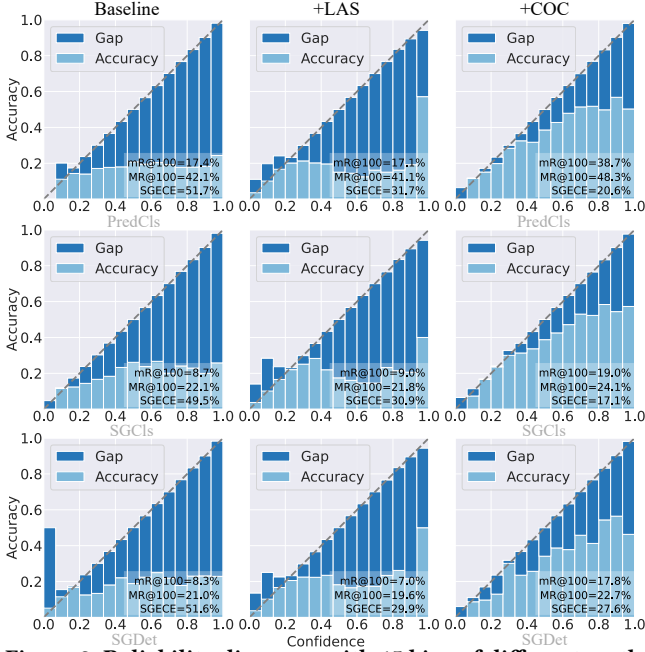


Figure 9: Reliability diagrams with 15 bins of different methods with Motifs [13] as baseline on the GQA-200 dataset.

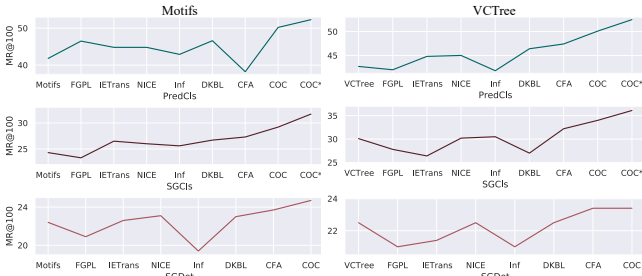


Figure 10: The MR@100 performance comparison with Motifs and VCTree as baselines on the VG dataset.

For sparse tail classes, CTDC fits poorly while LTDC fits nicely. We also present the tail class performance in Fig. 6. It is evident that LTDC more effectively improves the $tR@100$ of sparse tail classes. Moreover, as shown in Tab. 3, the better SGECE performance of LTDC demonstrates its superiority in improving calibration.

3) Effectiveness of CTR. In Tab. 4, we observe that the LTDC (w/o CTR) exhibits decreased $hR@K$ compared to the LTDC (w/ CTR). We argue that while LTDC enhances the performance of tail classes, there appear to be difficulties in effectively balancing performance across various classes, which is suboptimal for long-tailed SGG. To intuitively explore the effect of CTR on addressing this issue, we depict changes in confidence and recall in Fig. 7. It can be seen that adding CTR mitigates the extent of confidence degradation in head classes and further alleviates the decline in recall of head classes.

3 ADDITIONAL EXPERIMENTS ON CALIBRATION

To validate the generalization of COC for improving calibration, we present additional experimental results on calibration, including the calibration performance on VCTree baseline and the GQA dataset:

1) Experiments on VCTree baseline. As shown in Fig. 8, we report the calibration performance on the VCTree baseline. We have two observations: i) Compared to the baseline, COC effectively reduces SGECE and improves the $mR@K$ and $MR@K$, which demonstrates COC as a plug-and-play approach that can be applicable across different baselines, specializing in improving calibration and fostering more balanced learning. ii) Compared to specific calibration methods, e.g., FL (Focal Loss) [9] and LAS (Label Aware Smooth) [16], along with balanced learning methods, e.g., FGPL [7], IETrans [14] and NICE [4] methods, COC performs best on calibration (i.e., best SGECE) and balanced learning (i.e., best $MR@K$).

2) Experiments on GQA-200 dataset. As shown in Fig. 9, we report the calibration performance on the GQA-200 dataset. Compared to the baseline, COC achieves improvements in SGECE across three tasks. Moreover, COC also outperforms the LAS method [16]. These results demonstrate the effectiveness of COC in enhancing model calibration on the GQA-200 dataset, underscoring its generalizability when applied to different datasets.

4 ADDITIONAL ANALYSIS ON BALANCED LEARNING

1) Trade-off performance for long-tailed SGG. From Fig. 11, it can be observed that COC achieves the best trade-off performance between $R@100$ and $mR@100$ metrics across all three tasks. Additionally, when employing the ReINms trick, COC attains the best $R@100$ and $mR@100$ scores on both PredCls and SGCl tasks. We

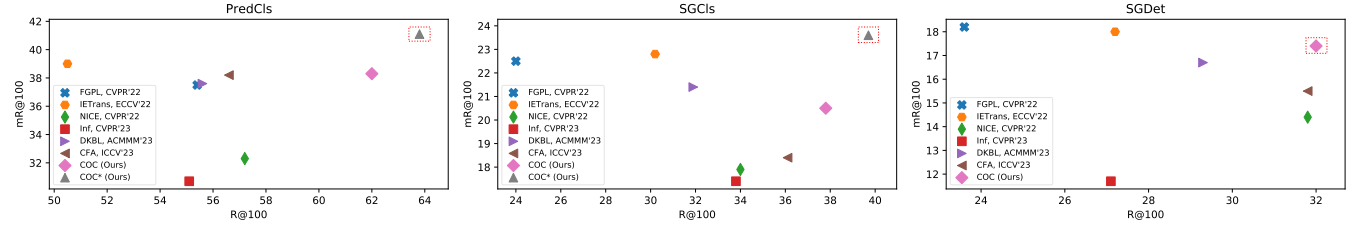


Figure 11: Illustration of the performance distribution of model-agnostic SGG models under the metrics of R@100 and mR@100. Models achieving the best MR@100 performance are highlighted with a red rectangle. The baseline model is Motifs [13]. COC* denotes the COC model using the RelNms trick [6, 15]. Due to the large number of predicted object pairs in SGDet task, applying pair-specific RelNms during evaluation can be time-consuming, so this trick is not utilized in SGDet task.

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
Baseline [11, 13] <small>CVPR '18</small>	65.3 / 67.2	14.9 / 16.3	40.1 / 41.8	38.9 / 39.8	8.3 / 8.8	23.6 / 24.3	32.1 / 36.8	6.6 / 7.9	19.4 / 22.4
+ FGFL [7] <small>CVPR '22</small>	51.5 / 55.4	33.0 / 37.5	42.3 / 46.5	23.4 / 24.0	21.3 / 22.5	22.4 / 23.3	20.8 / 23.6	15.4 / 18.2	18.1 / 20.9
+ IETrans [14] <small>ECCV '22</small>	48.6 / 50.5	35.8 / 39.0	42.2 / 44.8	29.4 / 30.2	21.5 / 22.8	25.5 / 26.5	23.5 / 27.2	15.5 / 18.0	19.5 / 22.6
+ NICE [4] <small>CVPR '22</small>	55.1 / 57.2	29.9 / 32.3	42.5 / 44.8	33.1 / 34.0	16.6 / 17.9	24.9 / 26.0	27.8 / 31.8	12.2 / 14.4	20.0 / 23.1
+ COC	59.7 / 62.0	35.3 / 38.3	47.5 / 50.2	36.8 / 37.8	19.6 / 20.5	28.2 / 29.2	27.6 / 32.0	15.1 / 17.4	21.4 / 24.7
+ COC + LA	53.3 / 56.0	39.4 / 42.6	46.4 / 49.3	33.1 / 34.0	22.2 / 24.4	27.7 / 29.2	25.2 / 29.5	16.3 / 19.0	20.8 / 24.3

Table 5: Performance comparison of various methods on VG. The baseline model is Motifs[13]. LA is the logit adjustment [8].

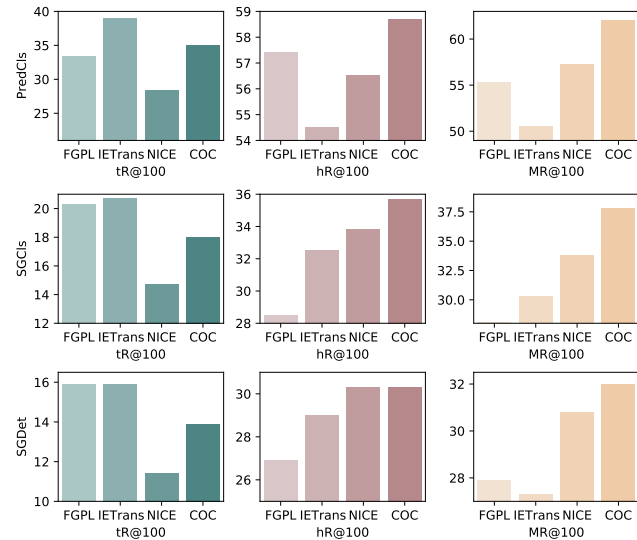


Figure 12: Performance comparison on different metrics among various methods. hR@K and tR@K denote the mean recall of head and tail groups, respectively.

also plot the MR@100 performance of various balanced-learning methods in Fig. 10. It is evident that COC performs the best on both Motifs and VCTree baselines, showcasing its most robust capability in promoting balanced learning for long-tailed SGG.

2) Additional analysis on the mR@K performance. We notice that the mR@K performance of COC is not as competitive as certain methods (e.g., IETrans [14]). This is because these methods tend to excessively sacrifice head-class performance to improve tail-class performance, which does not qualify as superb balancing learning. We display the performance of head and tail groups in Fig. 12 and notice that COC minimizes the decline in hR@100 and also performs best in MR@100. This showcases the excellent ability of COC to

strike a balance between improving tail-class performance without significantly compromising the performance of head classes.

In addition, we find that the mR@K performance can be improved through employing the simple logit adjustment method (i.e., LA [8]). As shown in Tab. 5, integrating the LA method into COC surpasses other methods on mR@K across three tasks, achieving higher R@K and MR@K as well. However, the MR@K does not improve and even declines compared to the COC in the PredCls and SGDet tasks. These results demonstrate two points: 1) solely focusing on improving mR@K may not necessarily lead to the optimal balanced learning method, as there may be excessive damage to the performance of majority classes; 2) while other methods may achieve better mR@K performance compared to COC, they actually inflict significant damage on the performance of majority classes, thereby leading to poor performance in R@K and MR@K, underscoring these methods not optimal balanced learning methods.

3) Predicate recall. Fig. 13 illustrates the R@100 performance of each relation using different methods. It is evident that PC, BC, and TDC contribute incrementally to enhancing the Recall@100 performance of tail classes. Furthermore, Motifs+PC+BC+TDC (i.e., the final proposed COC method) demonstrates the best performance across almost all tail classes compared to others, while maintaining a generally comparable level of recall performance on head classes with others. These results provide compelling evidence that COC can effectively alleviate biased predictions and excel at predicting fine-grained tail classes.

5 HYPERPARAMETER SELECTION

In this section, we study various hyperparameter strategies and report their respective performances. Within the COC method, the parameters that require adjustment primarily appear when employing the LTDC and the piecewise curriculum function in CTR. Specifically, the hyperparameters that require adjustment include β in LTDC, along with α and M in the piecewise curriculum function.

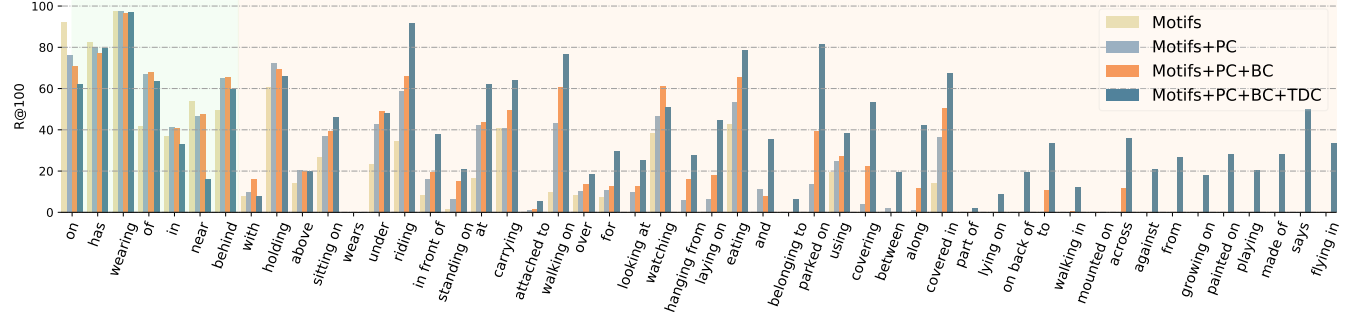


Figure 13: The R@100 performance for each class in the PredCls task on the VG dataset. We incrementally add each module to the Motifs [13] baseline. Relationship classes are sorted in descending order based on the number of samples.

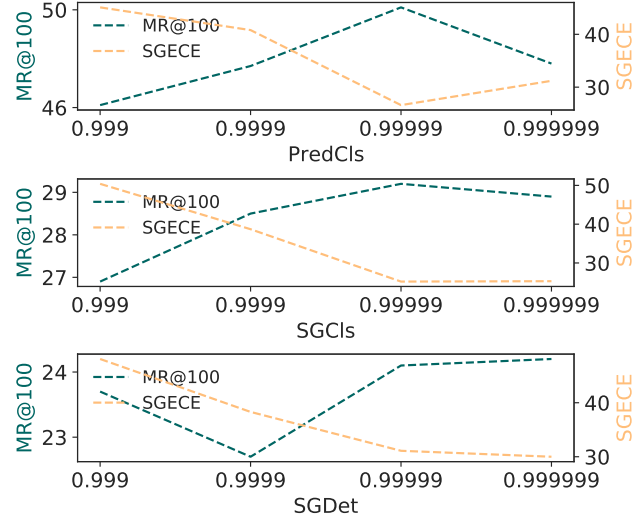


Figure 14: Influence of β on MR@100 and SGECE metrics.

stage	M	α	PredCls			
			R@50/100	mR@50/100	MR@50/100	SGECE
-	-	-	57.8 / 60.2	31.1 / 34.5	44.5 / 47.4	31.8
1	5	3/4	60.9 / 63.1	33.2 / 35.2	47.1 / 49.2	31.3
1	15	3/4	60.3 / 62.6	33.3 / 36.7	46.8 / 49.7	30.4
1	25	3/4	60.3 / 62.4	32.8 / 35.1	46.6 / 48.8	31.1
2	-	3/4	59.7 / 62.0	35.3 / 38.3	47.5 / 50.2	26.6
2	-	2/4	60.2 / 62.5	32.6 / 35.5	46.4 / 49.0	31.6
2	-	1/4	57.2 / 59.7	31.6 / 34.5	44.4 / 47.1	30.4

Table 6: The ablation study of different M and α values in the piecewise curriculum function.

1) Hyperparameter β in LTDC. The hyperparameter β ranges between 0 and 1. Since ϵ in LTDC is exponentially dependent on β , small variations in the value of β can have a significant impact. We conduct ablation studies on different values of β in proximity to 1 and report their performances in Fig. 14. It can be observed that for the PredCls and SGCl tasks, the MR@100 and SGECE are best with $\beta = 0.99999$; for the SgDet task, the MR@100 and SGECE are best with $\beta = 0.999999$. Therefore, we select $\beta = 0.99999$ for the PredCls and SGCl tasks and $\beta = 0.999999$ for the SgDet task.

2) Hyperparameter M and α in piecewise curriculum function in CTR. The hyperparameter M acts as an upper bound for ϵ during the first training stage. A smaller M indicates a narrower

range of acceptable ϵ and lower tolerance for the increased ϵ assigned to hard-to-learn tail classes in the first training stage. We experiment with different values of M in multiples of 5 and report their performances in Tab. 6. We observe that both MR@K and mR@K initially increase and then decrease. Finally, we choose $M = 15$ for its best performance on MR@K and SGECE.

The hyperparameter α determines the range of iterations of constrained learning for tail classes. A larger α shortens the second stage for assigning complete ϵ to tail classes, thereby indirectly extending the interval for focusing on learning head classes. We report the performances of three α values in Tab. 6, including 1/4, 1/2, and 3/4. It can be observed that with increasing α , both MR@K and mR@K improve. We choose $\alpha = 3/4$ as it exhibits the best performance on MR@K and SGECE.

Ultimately, compared to the baseline model without using CTR (reported in the first row), the addition of CTR achieves better R@K. This changing trend aligns with the designed principles of piecewise curriculum function in CTR, which emphasize reducing attention to hard-to-learn tail classes in the early training stage and concentrating on learning easy-to-learn head classes first. Furthermore, we observe an improvement in mR@K performance. We think that head classes often provide foundational interaction information. For example, head class ‘on’ actually shares the similar interaction pattern with tail classes ‘sitting on’, ‘standing on’ and so on. Therefore, early focused learning for head classes establishes a better understanding of foundational interaction patterns, which contributes to improved recognition of tail classes as well.

6 QUALITATIVE RESULTS

Miscalibrated models are unstable and pose safety risks in practical applications. To qualitatively assess the effectiveness of COC in mitigating miscalibration for long-tailed SGG, we collect images of road scenes commonly encountered in autonomous driving and analyze the qualitative results of these images. Specifically, we find COC can solve two issues contributing to miscalibration in long-tailed SGG: 1) Overconfidence in head classes, where even if the predicted head classes are incorrect, they may still obtain high confidence scores. For example, as shown in Fig.15 (a), the baseline model incorrectly predicts ‘man on motorcycle’ with high confidence, while the COC correctly predicts ‘man riding motorcycle’ with high confidence. 2) Underconfidence in tail classes, where certain tail classes may be correctly predicted but have relatively low confidence scores. For instance, as depicted in Fig.15 (c), the baseline model correctly

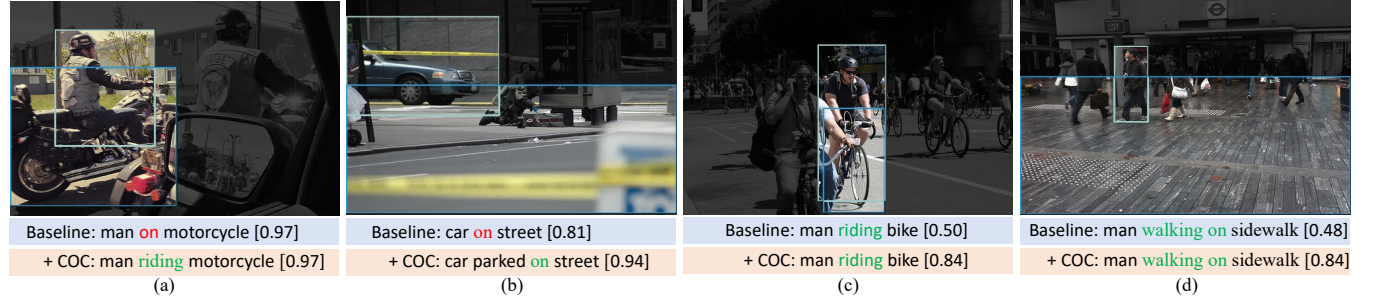


Figure 15: Qualitative comparisons between Baseline (i.e., Motifs [13]) and Baseline+COC are conducted in terms of Recall@50 on the PredCls task. The predicted classes correctly matching the ground truth are marked in green, while the incorrectly predicted classes are marked in red. The [confidence score] denotes the predicted confidence score of the detected relationship.

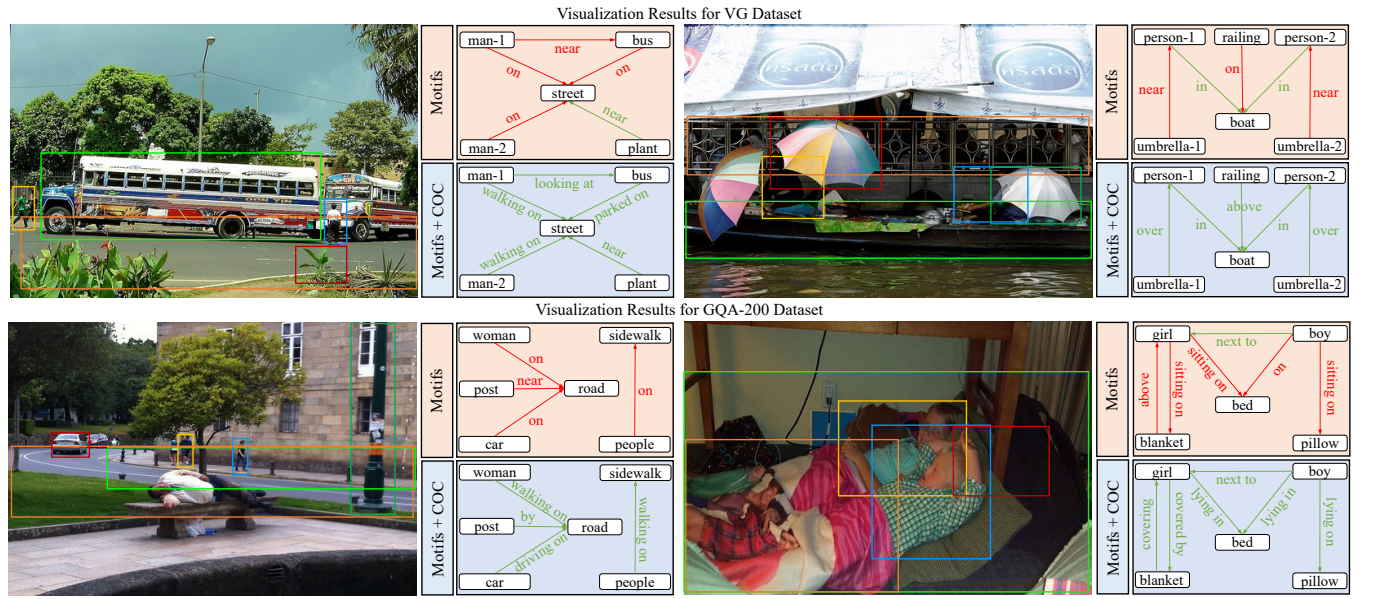


Figure 16: Qualitative comparisons between Motifs [13] and Motifs+COC are conducted in terms of Recall@50 on the PredCls task. Green edges indicate correctly predicted relationships matching the ground truth, while red solid edges indicate instances where the models fail to detect the ground truth relationships.

predicts ‘man riding bike’ but with a low confidence score. COC can correctly predict tail class ‘riding’ with a high confidence score.

Apart from qualitatively analyzing the effectiveness of COC in addressing miscalibration, we also qualitatively examine its ability to mitigate biased prediction. In Fig. 16, we showcase several examples derived from the Motifs [13], Motifs+COC models. In addition to the coarse-grained relationships (typically the majority classes [14]), COC can enable the baseline model to generate more fine-grained relationships (typically the minority classes). Specifically, the method COC can predict the right fine-grained relations (e.g., ‘looking at’ and ‘lying in’) rather than the wrong coarse-grained relations (e.g., ‘near’ and ‘on’). As shown in ‘man-looking-at-bus’ in the top-left scenario and ‘boy-lying-in-bed’ in the bottom-right scenario depicted in Fig. 16, these examples substantiate the effectiveness of COC in accurately predicting fine-grained relationships.

In conclusion, COC proves effective in helping produce well-calibrated and unbiased SGG models, thereby generating more trustworthy and fine-grained scene graphs, which can significantly bolster the utility of SGG models in facilitating downstream tasks.

REFERENCES

- [1] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. 2019. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9268–9277.
- [2] Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 6700–6709.
- [3] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision* 123, 1 (2017), 32–73.
- [4] Lin Li, Long Chen, Yifeng Huang, Zhimeng Zhang, Songyang Zhang, and Jun Xiao. 2022. The devil is in the labels: Noisy label correction for robust scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision*

- and Pattern Recognition. 18869–18878.
- [5] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2117–2125.
- [6] Xin Lin, Changxing Ding, Jing Zhang, Yibing Zhan, and Dacheng Tao. 2022. Ru-net: Regularized unrolling network for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19457–19466.
- [7] Xinyu Lyu, Lianli Gao, Yuyu Guo, Zhou Zhao, Hao Huang, Heng Tao Shen, and Jingkuan Song. 2022. Fine-grained predicates learning for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19467–19475.
- [8] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. 2020. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314* (2020).
- [9] Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. 2020. Calibrating deep neural networks using focal loss. *Advances in Neural Information Processing Systems* 33 (2020), 15288–15299.
- [10] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015).
- [11] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. 2020. Unbiased Scene Graph Generation From Biased Training. In *CVPR*.
- [12] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. 2019. Learning to Compose Dynamic Tree Structures for Visual Contexts. In *CVPR*.
- [13] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. 2018. Neural Motifs: Scene Graph Parsing With Global Context. In *CVPR*.
- [14] Ao Zhang, Yuan Yao, Qianyu Chen, Wei Ji, Zhiyuan Liu, Maosong Sun, and Tat-Seng Chua. 2022. Fine-Grained Scene Graph Generation with Data Transfer. In *ECCV*.
- [15] Chaofan Zheng, Xinyu Lyu, Lianli Gao, Bo Dai, and Jingkuan Song. 2023. Prototype-based Embedding Network for Scene Graph Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 22783–22792.
- [16] Zhisheng Zhong, Jiequan Cui, Shu Liu, and Jiaya Jia. 2021. Improving calibration for long-tailed recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16489–16498.