

Meta Weighted Loss: Balanced Scene Graph Generation with Meta-Learning

Yisen Wang¹, Yang Wang^{2*} and Yuxin Deng³

East China Normal University, Software Engineering Institute, Shanghai 200060, China

1 51255902105@stu.ecnu.edu.cn

2, 3 {ywang, yxdeng}@sei.ecnu.edu.cn

Abstract. Unbiased Scene Graph Generation (SGG) is a major direction of SGG. Recent years, a number of great methods have emerged in this field. But unfortunately, there is a critical conflict which is often unresolved between datasets, loss function and metrics: In most relevant datasets, the proportion of different predicates usually varies greatly. But each predicate has the same weight in loss function which can not properly reflect their unbalance in datasets. And when we evaluate results, this unbalanced issue also exists as we treat predicates in a uniform way. To mitigate this conflict, we introduce Meta Weighted Loss (MWL), a method based on meta-learning. MWL leverages meta-learning principles to construct a meta-neural network during model training. This network establishes a rational relationship between various predicates and their respective weight in loss function, so that alleviate the conflict. Furthermore, we also introduce a downsampling method named Neighbor Cleaning Rule (NCL), to build a more balanced sub-dataset which improve the training efficiency and performance during meta-training stage. Experimental results verify the effectiveness and generalization of MWL and NCL. Comprehensive experiments demonstrate that our method achieves the absolute gains up to 14.3% on VG dataset compared to baseline.

Keywords: Scene Graph Generation • Meta-Learning.

1 First Section

Scene Graph Generation (SGG) is a fundamental task in computer vision, aiming to detect entities in an image and the relationships between them, as shown in Fig. 1. The outcome of SGG can be represented as a graph, which provides a standardized and comprehensive representation of visual content in an image, serving as a solid foundation for downstream tasks such as Image Captioning [8,29] and Visual Question Answering (VQA) [9,13,21].

* indicates the corresponding author

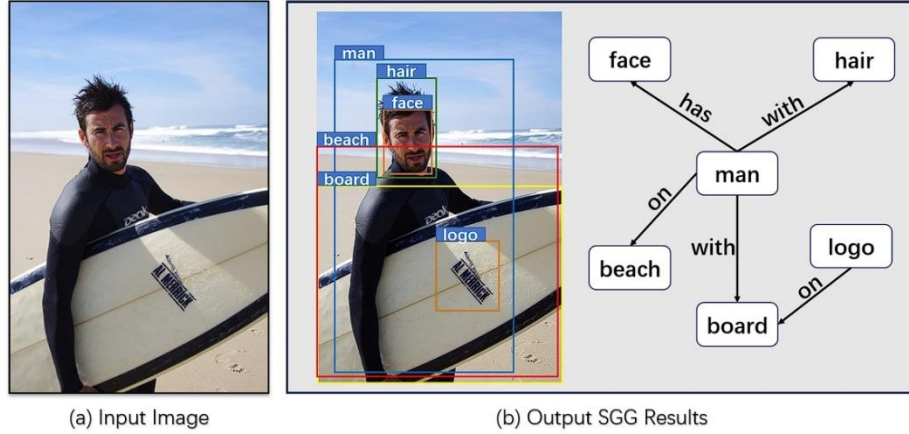


Fig. 1. Illustration of SGG. (a) An input image. (b) The output SGG results, including entities detected in the input image, corresponding labels, and relationships between entities.

In recent years, the field of SGG has witnessed remarkable progress, driven in part by the availability of large datasets like Visual Genome (VG) [12]. State-of-the-art models, including Motifs [32] and VcTree [23,25], have emerged, capable of producing

detailed scene graphs. However, a significant challenge arises when the only used recall metric may not be sufficient to meet the requirements of downstream tasks. The pursuit of high recall often leads to a dominance of common predicates (e.g., on, has) over more informative predicates (e.g., sitting on, carrying). Therefore, unbiased SGG has gradually become the mainstream research direction in the field of SGG, where the goal is to produce predictions that are not biased.

Unbiased SGG faces several challenges, with the long-tailed distribution of predicates in datasets being a prominent concern. As illustrated in Fig. 2, we can see that Recalls of common predicates are high but Recalls of uncommon predicates are extremely low. Numerous methods and modules have been dedicated to addressing this bias [4,10,14,24,30]. However, a key phenomenon often goes unnoticed: while unbiased SGG may produce balanced mean Recall metric, it does not penalize long-tailed predicates with extremely low scores. This can ultimately lead to a severe negative impact of dataset long-tail problems on the evaluation metric.

In tackling this challenge, our key insight is to adapt the weight assigned to different predicates in loss function. We present a framework based on meta-learning, consisting of the following key components:

Meta-Weight Network (MWN): The foundation of our method lies in MWN, initialized as a neural network. The primary function of MWN is to learn and assign appropriate weight to each predicate category.

Joint Training of Predicate Classifier and MWN: In the pursuit of balanced and unbiased predicate classification results, we emphasize the joint training of the

predicate classifier and MWN. This training mainly consists of two repeated processes: traditional training of predicate classifier and training of MWN. By verifying the effect of

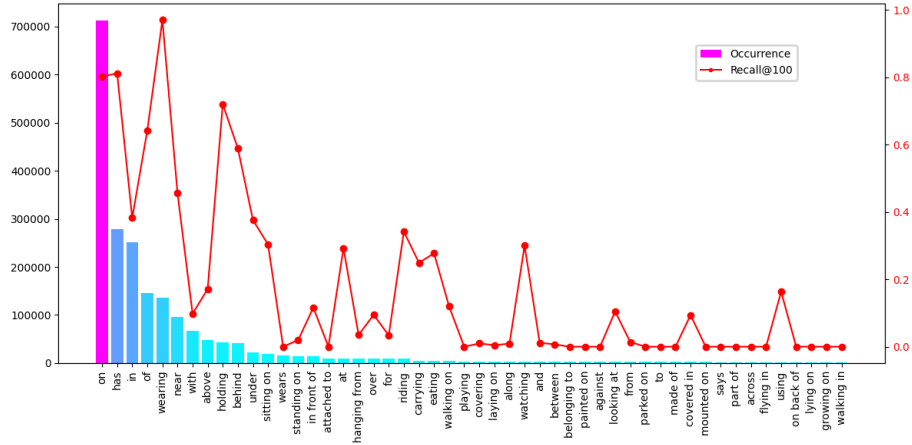


Fig. 2. Illustration of the long-tailed problem of VG [12] dataset and the unbalanced Recall of predicates. **Bar chart:** Represents the number of occurrences of 50 predicate categories in the dataset. **Line chart:** Represents the Recall@100 of the 50 predicate categories in Baseline model(*i.e.*, Motifs[32])’s prediction.

weights from MWN on an unbiased meta-validation set, the parameters of MWN are constantly adjusted to produce more appropriate weight. Then appropriate weight is applied to the classification loss during training of predicate classifier, which makes predicate classifier more and more unbiased.

In addition, in the joint training phase we mentioned a balanced meta-validation set. To get a representative meta-validation set, we leverage the Neighbor Cleaning Rule (NCL) method to downsample each predicate category. This results in a balanced dataset with more distinctive feature distributions across various predicates.

In summary, our work offers three primary contributions:

- We analyze and address the bias between predicate distribution in datasets and loss function within SGG, introducing the concept of using loss as a means to mitigate bias in unbiased SGG.
- We introduce NCL method to construct a balanced sub-dataset, promoting the inclusion of more representative predicate instances.
- We propose a meta-weight learning framework that leverages meta-learning principles to identify optimal predicate weight. In subtasks of SGG, our method significantly outperforms baseline models and achieves state-of-the-art results in many metrics.

2 Related Works

2.1 Scene Graph Generation

SGG is a critical task to understand scenes in computer vision, which has developed for a long time. It was first introduced by Lu *et al.* [20]. In the early stage, most methods focused on adding additional information and features from sources more than visual scene [6,18]. Later works involved incorporating spatial information into the feature extraction process. For instance, Zhu *et al.* [35] explored the significance of spatial distribution in object relationships, shedding light on the importance of spatial context. For another aspect, Baier *et al.* [2] introduced statistical priors by training models with absolute predicate frequencies, providing a valuable strategy to improve prediction accuracy. Subsequent advancements in SGG have focused on the utilization of visual context through sophisticated techniques. For example, [16] employed local message passing within triplets to refine object relationships. And some method proposed more powerful relation encoders with informational context, like sequential LSTMs [25,32].

2.2 Unbiased Scene Graph Generation

Recent years, after the introduction of the less biased mean recall metric by Chen *et al.* [3] and Tang *et al.* [25], the SGG researchers started to pay attention to the imbalance problem in SGG. Then unbiased SGG became a mainstream direction in the field of SGG and various methods was proposed to solve the bias problem. As more and more methods emerged, two main categories of approaches can be summarized in the field of unbiased SGG, as summarized in [14]: 1) After Tang *et al.* [24] proposed that causal reasoning solves predictive bias based on two strong SGG models [25,32]. Researchers consider Motifs [32] and VCTree [25] to be the two most commonly used baselines and propose various model-agnostic methods on these baselines [4,10,14,24,28,33]. 2) Besides model-agnostic methods, there is also another efficiency way to improve unbiased SGG. That is to create special-designed models instead of baselines to achieve SGG tasks [7,15,17,19,31,34]. Li *et al.* [15] employed a re-sampling strategy while Zheng *et al.* [34] proposed a new prediction model based on prototype embedding. Within the first category towards unbiased SGG, many methods will decline Recall metric because of the improvement strategy on mean Recall. And they usually overlook the conflict we mentioned so that these unbiased model-agnostic methods can't make a comprehensive improvement on SGG. Our method mitigates the conflict and while improving mean Recall metric, the decrease of Recall metric is minimized to the greatest extent, thus enhancing SGG performance more comprehensively.

2.3 Meta-Learning

Meta-learning, also known as learning to learn, has emerged as a powerful paradigm for adapting models to new tasks, particularly when faced with limited data [11,26]. In

the context of SGG, meta-learning offers a compelling method to address challenges such as biased loss and model efficiency. Ren *et al.* [25] made pioneering contributions by introducing a meta-learning framework to reweight unbalanced samples. Similarly, Shu *et al.* [22] harnessed multi-layer perceptrons (MLPs) to learn weight functions for reweighting, highlighting the potential of meta-learning in rebalancing SGG models.

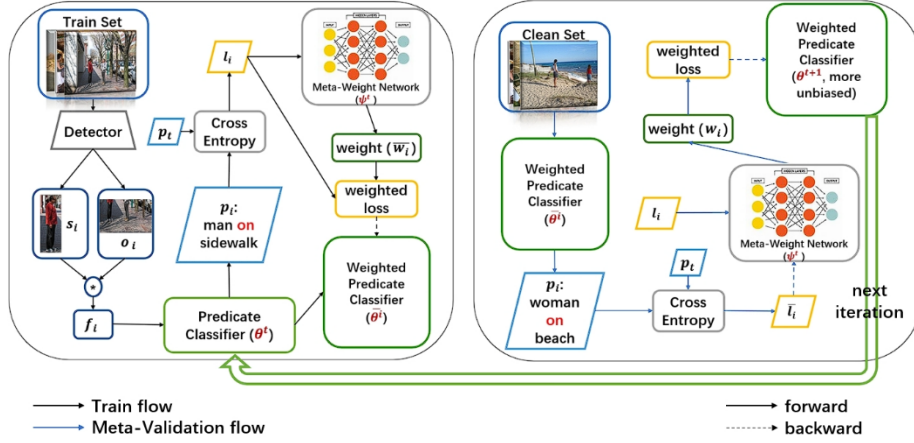


Fig. 3. The pipeline of our method MWL. During one iteration: **Left:** the routine training of Predicate Classifier with θ^t as parameter; **Right:** the process of updating the Meta-Weight Network with ψ^t as parameter, by training on the clean dataset, and then obtaining a more unbiased Predicate Classifier as the Predicate Classifier in the next iteration. $\bar{*}$ denotes the intermediate variable in one iteration.

3 Method

3.1 Pipeline description

The whole pipeline of applying Meta Weighted Loss (MWL) in SGG is illustrated in Fig.3. To illustrate, consider a specific scenario: 1) In a conventional training set, the predicate *on* occurs far more frequently than *standing on*, causing the Predicate Classifier with parameter θ^t to be strongly biased toward predicting *on* rather than *standing on*. 2) MWN with parameter ψ^t generates a random initial weight \bar{w}_1 based on the training loss l_i . However, applying this weight may not guarantee that the Weighted Predicate Classifier $\bar{\theta}^t$ is less biased. Because at this time, MWN has not been trained with the balanced meta-validation set. 3) Consequently, we assess the performance of $\bar{\theta}^t$ on a balanced meta-validation dataset, we call it "Clean Set", which does not exhibit serious long-tailed problems. Specifically, the number of instances of *on* and *standing on* is approximately equal, and the proportion of each predicate category is nearly balanced. Predictions on Clean Set may still favor *on* over *standing on*, and its corresponding loss \bar{l}_1 can be used to update ψ^t . 4) After ψ^t is updated to ψ^{t+1} ,

l_i is once again input into MWN, which generates an updated weight w_i . This updated weight helps make the Weighted Predicate Classifier more unbiased by reducing the weight of *on* and increasing the weight of *standing on* in subsequent predictions. In this process, MWN serves as a tool for learning how to train Predicate Classifier more effectively by adapting to Clean Set, exemplifying the application of meta-learning in SGG.

3.2 Meta Weighted Loss Learning

Let $f_i = \langle s_i, o_i \rangle$ denotes the feature embedding of the i -th instance feeding into Predicate Classifier, where s_i is the feature embedding of i -th subject and o_i is the feature embedding of i -th object. The training set is denoted as T and Clean Dataset for meta-validation is denoted as C . As illustrated in Fig. 3, θ^t is the parameter of Predicate Classifier. After f_i is input into Predicate Classifier, predicate prediction result of this subject-object pair is obtained, and then Cross Entropy loss l_i is calculated according to ground-truth. In the normal training process, the optimal Predicate Classifier parameters can be obtained by minimizing the following loss:

$$L(\theta) = \frac{1}{n} \sum_{i=1}^n l_i = \frac{1}{n} \sum_{i=1}^n \left(-\frac{1}{C} \sum_{c=1}^n p_c \log p_{i,c} \right), (1)$$

where n is the number of training instances in a batch and C is the number of predicate categories. p_c is the ground-truth predicate and $p_{i,c}$ is the predicate prediction of current instance.

In our pipeline, MWN will take loss l_i as input and output corresponding intermediate weight $\bar{w}_i(\psi)$. After applying $\bar{w}_i(\psi)$ to Predicate Classifier's loss l_i , the optimal Weighted Predicate Classifier can be obtained by minimizing the following loss:

$$\begin{aligned} L^{weighted}(\theta) &= \frac{1}{n} \sum_{i=1}^n \bar{w}_i(\psi) \cdot l_i(\theta) \\ &= \frac{1}{n} \sum_{i=1}^n \left(-\frac{\bar{w}_i(\psi)}{C} \sum_{c=1}^n p_c \log p_{i,c} \right). (2) \end{aligned}$$

So, at t -th iteration, $\bar{\theta}^t$ will be updated:

$$\begin{aligned} \bar{\theta}^t &= \theta^t - \eta_1 \nabla_{\theta^t} L^{weighted}(\theta^t)|_{\theta^t} \\ &= \theta^t - \eta_1 \frac{1}{n} \sum_{i=1}^n \nabla_{\theta^t} \bar{w}_i(\psi^t) \cdot l_i(\theta^t)|_{\theta^t}, (3) \end{aligned}$$

where η_1 is the learning rate of Predicate Classifier. This updated $\bar{\theta}^t$ is regarded as intermediate parameter since its weight \bar{w}_i hasn't been updated through meta-validation process.

The second step is meta-validation process. We validate Weighted Predicate Classifier (with $\bar{\theta}^t$ as parameter) on the clean dataset, and the validation loss is denoted as \bar{l}_t . Then we can update the parameter ψ^t of MWN through \bar{l}_t in the t -th iteration:

$$\psi^{t+1} = \psi^t - \eta_2 \frac{1}{n} \sum_{i=1}^n \nabla_{\psi^t} L(\bar{\theta}^t)|_{\psi^t}, \quad (4)$$

where η_2 is the learning rate of MWN.

Finally, after ψ^t is updated into ψ^{t+1} , we feed loss l_i of training into MWN again to obtain a new weight w_i . Then we apply this w_i to loss l_i . Based on this new weighted loss, Weighted Predicate Classifier is updated again in a similar manner to Eq. 3:

$$\theta^{t+1} = \theta^t - \eta_1 \frac{1}{n} \sum_{i=1}^n \nabla_{\theta^t} \bar{w}_i(\psi^{t+1}) \cdot l_i(\theta^t)|_{\theta^t}. \quad (5)$$

3.3 Neighbor Cleaning Rule

Neighbor Cleaning Rule (NCL)[1,5] primarily conducts data cleaning based on neighbor relationships, determining whether an instance should be retained by assessing its similarity to neighboring instances. In our context, instance similarity is gauged using the distance between feature vectors, and we employ the square of the Euclidean distance as the similarity measure:

$$d(f_a, f_b) = \|f_a - f_b\|_2, \quad (6)$$

where f_a represents the feature embedding corresponding to the triplet instance a , like f_i at the beginning of Sec. 3.1, and $d(f_a, f_b)$ represents the Euclidean distance between instances a and b .

Our downsampling process mainly consists of the following steps:

1) For the data points contained in class c , we calculate data density of central region. Let μ_c denote the cluster center of class c , then we can obtain a distances set $\mathbf{G} = \{g_i\}_{i=0}^N$, where N is the total number of points belonging to class c and g_i is computed as:

$$g_i = d(f_i, \mu_c). \quad (7)$$

Then, we sort this set and remove the largest 10% of elements to obtain the remained distance set as $R = \{b_i\}_{i=0}^{0.9N}$, in order to eliminate the impact of overly remote data. Furthermore, the boundary distance D_c of class c can be calculated:

$$D_c = \alpha \frac{1}{0.9N} \sum_{i=1}^{0.9N} b_i, \quad (8)$$

where α is a hyper-parameter to adjust the distance used next. After we get D_c , the number of data points within distance D_c of μ_c can be obtained as k_c , which will be utilized as threshold in later steps.

2) For each data point f_i , compute the number of data points s_i within a distance not exceeding D_i around x_i :

$$s_i = \sum_{j=1}^N \delta(d(x_i, x_j) \leq D_i), \quad (9)$$

where $\delta(\cdot)$ is the indicator function. In s_i , the number of data points of class c is denoted as m_i .

3) If $m_i < \beta \cdot k_c$, the point is considered to be too far away from the class c , and delete it; else if $m_i < \gamma \cdot s_i$, it is considered that the point is in the confusion zone of different categories, and it is deleted; otherwise, the data point is retained. β and γ are hyper-parameters.

4 Experiments

4.1 Datasets

Visual Genome (VG). The VG dataset boasts a collection of 108,000 images, each annotated with an average of 38 objects and 22 relationships per image. In line with prior research efforts [4,14,24,30], we adopt the widely recognized split [27]. This split encompasses the most prevalent 150 object categories and 50 predicate categories. Furthermore, the dataset is thoughtfully partitioned into three sets: a training set (comprising 70% of the data), a test set (encompassing 30% of the data), and a validation set (comprising 5,000 images) extracted from the training set for model validation.

The Clean Dataset. As elucidated in Sec.3.2, we apply the Neighbor Cleaning Rule (NCL) method to downsample the VG dataset, resulting in a smaller, predicate-balanced dataset suitable for meta-validation. In our experiments, the clean dataset encompasses 5,000 images, with each predicate categories making approximately 400 appearances.

4.2 Tasks and Metrics

Tasks. We evaluated our method on three SGG sub-tasks [27]: 1) Predicate Classification (**PredCls**): Given the ground-truth entities with labels, we need to only predict pairwise predicate categories. 2) Scene Graph Classification (**SGCls**): Given the ground-truth entities bounding boxes, first we need to predict the entity categories, and then predict predicate categories. 3) Scene Graph Generation (**SGGen**): Given an image, we need to detect all bounding boxes of entities, and predict both the entity categories and predicate categories between entities.

Metrics. Following the previous works [2,4,10,14,24,30,34], we adopt three metrics as the primary evaluation metrics: 1) Recall@K (**R@K**): It calculates the proportion of

the top-K confident triplets in the ground-truth. In VG dataset, R@K is more representative of the predictions of common predicates because of the imbalanced data distribution. 2) mean Recall@K (**mR@K**): It calculates the recall of each predicate category separately, and then averages R@K over all predicate categories. In contrast to R@K, mR@K prefers tail predicates. 3) Mean@K (**M@K**): As introduced in [34], it averages the R@K and mR@K for evaluating the model's overall performance on SGG. We believe that this metric is more in line with the future development direction of SGG.

Table 1. Comparison between our method (MWL) and previous methods. The **best** and **second best** methods under each setting are marked according to formats. Category A is indicated to special-designed models and B is model-agnostic methods.

Category	Method	PredCls			SGCls			SGDet		
		R@50/100	mR@50/100	M@50/100	R@50/100	mR@50/100	M@50/100	R@50/100	mR@50/100	M@50/100
A	MSDN [17]	64.6 / 66.6	15.9 / 17.5	40.3 / 42.1	38.4 / 39.8	9.3 / 9.7	23.9 / 24.8	31.9 / 36.6	6.1 / 7.2	19.0 / 21.9
	GB-Net [31]	66.6 / 68.2	22.1 / 24.0	44.4 / 46.1	37.3 / 38.0	12.7 / 13.4	25.0 / 25.7	26.3 / 29.9	7.1 / 8.5	16.7 / 19.2
	BGNN [15]	59.2 / 61.3	30.4 / 32.9	44.8 / 47.1	37.4 / 38.5	14.3 / 16.5	25.9 / 27.5	31.0 / 35.8	10.7 / 12.6	20.9 / 24.2
	DT2-ACBS[7]	23.3 / 25.6	35.9 / 39.7	29.6 / 32.7	16.2 / 17.6	24.8 / 27.5	20.5 / 22.6	15.0 / 16.3	22.0 / 24.4	18.5 / 20.4
	PE-Net [34]	64.9 / 67.2	31.5 / 33.8	48.2 / 50.5	39.4 / 40.7	17.8 / 18.9	28.6 / 29.8	30.7 / 35.2	12.4 / 14.5	21.6 / 24.9
	B	Motifs [32]	65.5 / 67.2	16.5 / 17.8	41.0 / 42.5	39.0 / 39.7	8.7 / 9.3	23.9 / 24.5	32.1 / 36.9	5.5 / 6.8
Motifs+TDE [24]		45.0 / 50.6	24.2 / 27.9	34.6 / 39.3	27.1 / 29.5	13.1 / 14.9	20.1 / 22.2	17.3 / 20.8	9.2 / 11.1	13.3 / 16.0
Motifs+CogTree [30]		35.6 / 36.8	26.4 / 29.0	31.0 / 32.9	21.6 / 22.2	14.9 / 16.1	18.3 / 19.2	20.0 / 22.1	10.4 / 11.8	15.2 / 17.0
Motifs+DLFE [4]		52.5 / 54.2	26.9 / 28.8	39.7 / 41.5	32.3 / 33.1	15.2 / 15.9	23.8 / 24.5	25.4 / 29.4	11.7 / 13.8	18.6 / 21.6
Motifs+BPL-SA [10]		50.7 / 52.5	29.7 / 31.7	40.2 / 42.1	30.1 / 31.0	16.5 / 17.5	23.3 / 24.3	23.0 / 26.9	13.5 / 15.6	18.3 / 21.3
Motifs+NICE [14]		55.1 / 57.2	<u>29.9 / 32.3</u>	42.5 / 44.8	33.1 / 34.0	16.6 / 17.9	<u>24.9 / 26.0</u>	27.8 / 31.8	12.2 / 14.4	<u>20.0 / 23.1</u>
Motifs+IETrans [33]		54.7 / 56.7	30.9 / 33.6	<u>42.8 / 45.2</u>	32.5 / 33.4	<u>16.8 / 17.9</u>	24.7 / 25.7	26.4 / 30.6	12.4 / 14.9	19.4 / 22.8
Motifs+MWL(Ours)		<u>57.7 / 60.1</u>	28.6 / 30.9	43.2 / 45.5	<u>36.7 / 37.6</u>	17.2 / 18.1	27.0 / 27.9	<u>28.7 / 32.9</u>	<u>12.6 / 15.3</u>	20.7 / 24.1
VCTree [25]		65.9 / 67.5	17.1 / 18.4	41.5 / 43.0	45.6 / 46.5	10.8 / 11.5	28.2 / 29.0	32.0 / 36.2	7.2 / 8.4	19.6 / 22.3
VCTree+TDE [24]		44.8 / 49.2	26.2 / 29.6	35.5 / 39.4	28.8 / 32.0	15.2 / 17.5	22.0 / 24.8	17.3 / 20.9	9.5 / 11.4	13.4 / 16.2
VCTree+CogTree [30]		44.0 / 45.4	27.6 / 29.7	35.8 / 37.6	30.9 / 31.7	18.8 / 19.9	24.9 / 25.8	18.2 / 20.4	10.4 / 12.1	14.3 / 16.3
VCTree+DLFE [4]		51.8 / 53.5	25.3 / 27.1	38.6 / 39.4	33.5 / 34.6	18.9 / 20.0	26.2 / 27.3	22.7 / 26.3	11.8 / 13.8	17.3 / 20.1
VCTree+BPL-SA [10]		50.0 / 51.8	30.6 / 32.6	40.3 / 42.2	34.0 / 35.0	20.1 / 21.2	27.1 / 28.1	21.7 / 25.5	13.5 / 15.7	17.6 / 20.6
VCTree+NICE [14]		55.0 / 56.9	<u>30.7 / 33.0</u>	<u>42.9 / 45.0</u>	37.8 / 39.0	<u>19.9 / 21.3</u>	<u>28.9 / 30.2</u>	27.0 / 30.8	<u>11.9 / 14.1</u>	<u>19.5 / 22.5</u>
VCTree+IETrans [33]		53.0 / 55.0	30.3 / 33.9	41.7 / 44.5	32.9 / 33.8	16.5 / 18.1	24.7 / 26.0	25.4 / 29.3	11.5 / 14.0	18.5 / 21.7
VCTree+MWL(Ours)		<u>58.0 / 59.8</u>	30.9 / 32.7	44.5 / 46.3	<u>40.9 / 42.1</u>	19.8 / 20.8	30.4 / 31.5	<u>28.8 / 32.9</u>	11.8 / <u>14.1</u>	20.3 / 23.5

4.3 Comparisons with State-of-the-art Methods

To assess the SGG improvement of our method, in this section we mainly use MWL on two baselines: **Motifs** [32] and **VCTree** [25]. We compare it with several state-of-the-art SGG methods on VG dataset under all three sub-tasks. Based on the type of generalization of these methods, we divide them into the following two categories: 1) **MSDN** [17], **GB-Net** [31], **BGNN** [15], and **PE-Net** [34]. These methods are designed specifically for SGG which are not based on common baselines. 2) **TDE** [24], **CogTree** [30], **DLFE** [4], **NICE** [14] and **IETrans** [33]. These methods are model-agnostic SGG unbiased methods that are typically applied to baselines, e.g., **Motifs** [32] and **VCTree** [25]. The results are shown in Tab. 1.

Generally, our method achieved significant improvements on two strong baselines (i.e., **Motifs** [32], **VCTree** [25]). The absolute gains on metric mR@50 and mR@100

are 7.1% ~ 12.1% and 8.5% ~ 13.1% over Motifs, and 4.6% ~ 13.8%, 5.7% ~ 14.3% over VCTree. The results demonstrate the effectiveness of our method.

Due to the structure of datasets, any method to increase mR@K will inevitably reduce the R@K. Based on this, we believe that the goal of unbiased SGG should not only focus on improving mR@K, while ignoring the adverse effects on R@K. Therefore, M@K proposed in [34], which calculated from the average value of R@K and mR@K, should become an important metric in this field. Compared to other state-of-the-art model-agnostic debiasing strategies, MWL can not only achieve top performance on mR@K metrics, but also keep relatively high performance on R@K metrics, as the performance of our method on R@K is only inferior to the baseline. Finally, on M@K metrics which represents the comprehensive level of R@K and mR@K, our MWL always achieve state-of-the-art.

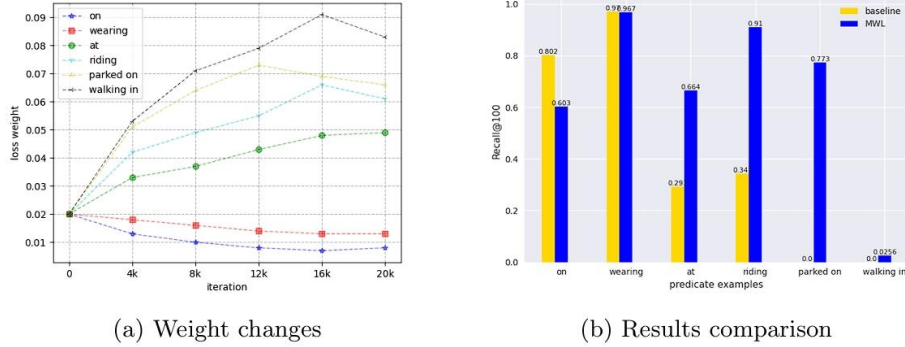


Fig. 4. The Illustration of weights changes and results comparison from predicates under different frequency level.

4.4 Specific Interpretation on How MWL Promote SGG

In order to better interpret our method, we recorded the changes of weight corresponding to predicates in joint training phase, and the results obtained by these predicates after changing their weight, as shown in Fig.4. For the convenience of illustration and considering the different levels of predicate frequency, we divide all predicates into three levels, and select 'on' and 'wearing' for common predicates, 'at' and 'riding' for predicates with the middle frequency, and 'parked on' and 'walking in' for rare predicates.

It is not difficult to see from Fig.4a that, as expected, the loss weight corresponding to common predicates are reduced, while the weight corresponding to medium-frequency and rare predicates are increased in the meta-validation stage. These changes reduce the dominance of common predicates and make the model pay more attention to the prediction of uncommon predicates. As reflected in Fig.4b, after this weight change, the Recalls of uncommon predicates are improved, and the Recalls of common predicates

are not reduced to a large extent, thus achieving an overall performance improvement of the SGG model.

Table 2. Ablation studies on components of MWL. The baseline model is Motifs [32].

Exp	Components		PredCls		
	MWN	NCL	mR@50/100	R@50/100	M@50/100
1	×	×	16.5 / 17.8	65.5 / 67.2	41.0 / 42.5
2	√	×	26.3 / 28.5	56.1 / 57.2	41.2 / 43.6
3	√	√	28.6 / 30.9	57.7 / 60.1	43.2 / 45.5

4.5 Ablation Studies

To assess the efficacy of the components in our method, we conduct ablation studies using VG dataset. The results are summarized in Tab. 2. In this table, Exp1 represents the baseline model. Notably, when we introduce MWN to the baseline model in Exp2, there is a substantial increase in metric mR@K. However, it is worth noting that R@K experiences a decrease, which is an anticipated trade-off when employing unbiased methods. In Exp3, we construct Clean Dataset using NCL method based on the improvements from Exp2. This not only further enhances mR@K but also mitigates the decline in R@K, ultimately allowing the metric M@K to achieve state-of-the-art performance. These ablation experiments verify the significant impact of MWN on the SGG tasks.

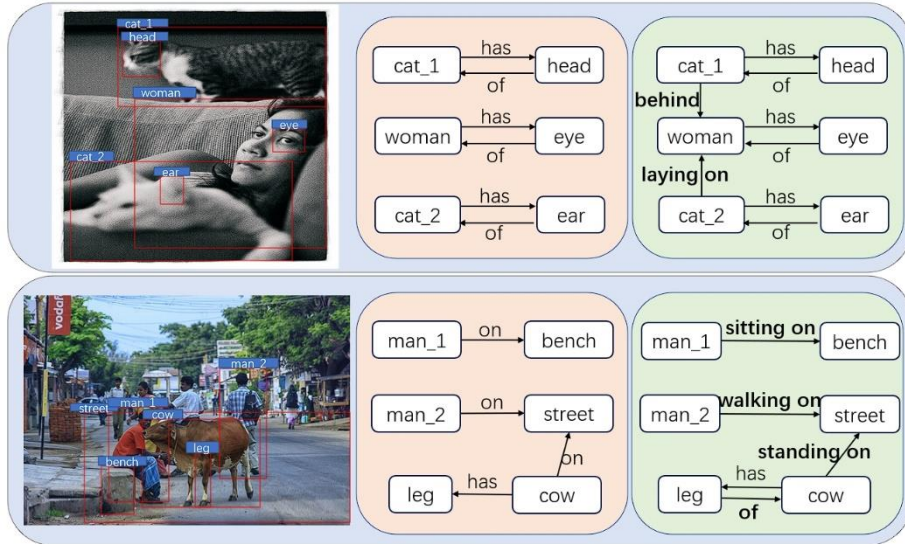


Fig. 5. Visualization SGG result of Motifs [32] (in orange) and MWL (in green) on the PredCls task. The predictions in bold indicate that our method makes the model more unbiased and the predictions more comprehensive and accurate.

4.6 Visualization Results

In order to illustrate MWL’s improvements over baseline more prominently, we present a comparative analysis of scene graph detection results generated by Motifs [32] and our method in Figure 5.

In the first example, our method predicts more relations, like “*cat_1-behind - women*” and “*cat_2-laying on-women*.” In the second example, our method predicts relations like, “*man 1-sitting on-bench*,” and “*cow-standing on-street*,” as opposed to the simpler “*man-on-bench*,” and “*cow-on-street*” provided by Motifs [32]. These results demonstrate the considerable effectiveness of our method, which contribute to a comprehensive understanding of the scene.

5 Conclusion

In this paper, we have firstly analyzed the conflict between datasets bias, loss function and pursuit metrics. Based on this, we have introduced a meta-learning framework and proposed Meta Weighted Loss to enhance baseline training and augment its overall performance. Furthermore, we have employed the Neighbor Cleaning Rule method within the meta-learning process to obtain a cleaner and more balanced meta-validation set to improve the training of model. Through extensive experiments, we have confirmed the effectiveness of MWL on the baselines. Among model-agnostic methods, MWL achieves new state-of-the-art performance on metric M@K, a comprehensive measure of model performance.

References

1. Agustianto, K., Destarianto, P.: Imbalance data handling using neighborhood cleaning rule (ncl) sampling method for precision student modeling. In: 2019 International Conference on Computer Science, Information Technology, and Electrical Engineering (ICOMITEE). pp. 86–89 (2019). <https://doi.org/10.1109/ICOMITEE.2019.8921159>
2. Baier, S., Ma, Y., Tresp, V.: Improving visual relationship detection using semantic modeling of scene descriptions. In: The Semantic Web–ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21–25, 2017, Proceedings, Part I 16. pp. 53–68. Springer (2017)
3. Chen, T., Yu, W., Chen, R., Lin, L.: Knowledge-embedded routing network for scene graph generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6163–6171 (2019)

4. Chiou, M.J., Ding, H., Yan, H., Wang, C., Zimmermann, R., Feng, J.: Recovering the unbiased scene graphs from the biased ones. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 1581–1590 (2021)
5. Choirunnisa, S., Meidyani, B., Rochimah, S.: Software defect prediction using oversampling algorithm: A-suwo. In: 2018 Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS). pp. 337–341 (2018). <https://doi.org/10.1109/EECCIS.2018.8692874>
6. Dai, B., Zhang, Y., Lin, D.: Detecting visual relationships with deep relational networks. In: Proceedings of the IEEE conference on computer vision and Pattern recognition. pp. 3076–3086 (2017)
7. Desai, A., Wu, T.Y., Tripathi, S., Vasconcelos, N.: Learning of visual relations: The devil is in the tails. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15404–15413 (2021)
8. Fei, J., Wang, T., Zhang, J., He, Z., Wang, C., Zheng, F.: Transferable decoding with visual entities for zero-shot image captioning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3136–3146 (2023)
9. Fu, X., Zhang, S., Kwon, G., Perera, P., Zhu, H., Zhang, Y., Li, A.H., Wang, W.Y., Wang, Z., Castelli, V., et al.: Generate then select: Open-ended visual question answering guided by world knowledge. arXiv preprint arXiv:2305.18842 (2023)
10. 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: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16383–16392 (2021)
11. Hospedales, T., Antoniou, A., Micaelli, P., Storkey, A.: Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence* 44(9), 5149–5169 (2021)
12. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision* 123, 32–73 (2017)
13. Li, K., Vosselman, G., Yang, M.Y.: Hrvqa: A visual question answering benchmark for high-resolution aerial images. arXiv preprint arXiv:2301.09460 (2023)
14. Li, L., Chen, L., Huang, Y., Zhang, Z., Zhang, S., Xiao, J.: 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. pp. 18869–18878 (2022)
15. Li, R., Zhang, S., Wan, B., He, X.: Bipartite graph network with adaptive message passing for unbiased scene graph generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11109–11119 (2021)
16. Li, Y., Ouyang, W., Wang, X., Tang, X.: Vip-cnn: Visual phrase guided convolutional neural network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1347–1356 (2017)
17. Li, Y., Ouyang, W., Zhou, B., Wang, K., Wang, X.: Scene graph generation from objects, phrases and region captions. In: Proceedings of the IEEE international conference on computer vision. pp. 1261–1270 (2017)
18. Liao, W., Rosenhahn, B., Shuai, L., Ying Yang, M.: Natural language guided visual relationship detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 0–0 (2019)

19. Lin, X., Ding, C., Zeng, J., Tao, D.: Gps-net: Graph property sensing network for scene graph generation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 3746–3753 (2020)
20. Lu, C., Krishna, R., Bernstein, M., Fei-Fei, L.: Visual relationship detection with language priors. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. pp. 852–869. Springer (2016)
21. Malinowski, M., Doersch, C., Santoro, A., Battaglia, P.: Learning visual question answering by bootstrapping hard attention. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. pp. 3–20 (2018)
22. Shu, J., Xie, Q., Yi, L., Zhao, Q., Zhou, S., Xu, Z., Meng, D.: Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems* 32 (2019)
23. Tang, K.: A scene graph generation codebase in pytorch (2020), [bluehttps://github.com/KaihuaTang/Scene-Graph-Benchmark.pytorch](https://github.com/KaihuaTang/Scene-Graph-Benchmark.pytorch)
24. Tang, K., Niu, Y., Huang, J., Shi, J., Zhang, H.: Unbiased scene graph generation from biased training. In: *Conference on Computer Vision and Pattern Recognition (2020)*
25. Tang, K., Zhang, H., Wu, B., Luo, W., Liu, W.: Learning to compose dynamic tree structures for visual contexts. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 6619–6628 (2019)
26. Vanschoren, J.: Meta-learning: A survey. *arXiv preprint arXiv:1810.03548* (2018)
27. Xu, D., Zhu, Y., Choy, C.B., Fei-Fei, L.: Scene graph generation by iterative message passing. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 5410–5419 (2017)
28. 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: *Proceedings of the 28th ACM international conference on multimedia*. pp. 265–273 (2020)
29. Yang, D., Chen, H., Hou, X., Ge, T., Jiang, Y., Jin, Q.: Visual captioning at will: Describing images and videos guided by a few stylized sentences. *arXiv preprint arXiv:2307.16399* (2023)
30. Yu, J., Chai, Y., Wang, Y., Hu, Y., Wu, Q.: Cogtree: Cognition tree loss for unbiased scene graph generation. *arXiv preprint arXiv:2009.07526* (2020)
31. Zareian, A., Karaman, S., Chang, S.F.: Bridging knowledge graphs to generate scene graphs. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*. pp. 606–623. Springer(2020)
32. Zellers, R., Yatskar, M., Thomson, S., Choi, Y.: Neural motifs: Scene graph parsing with global context. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 5831–5840 (2018)
33. Zhang, A., Yao, Y., Chen, Q., Ji, W., Liu, Z., Sun, M., seng Chua, T.: Fine-grained scene graph generation with data transfer. In: *European Conference on Computer Vision (2022)*, [bluehttps://api.semanticscholar.org/CorpusID:247596771](https://api.semanticscholar.org/CorpusID:247596771)
34. Zheng, C., Lyu, X., Gao, L., Dai, B., Song, J.: Prototype-based embedding network for scene graph generation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 22783–22792 (2023)
35. Zhu, Y., Jiang, S., Li, X.: Visual relationship detection with object spatial distribution. In: *2017 IEEE International Conference on Multimedia and Expo (ICME)*. pp. 379–384 (2017). <https://doi.org/10.1109/ICME.2017.8019448>