# YOLO-SG: An Efficient Framework for Scene Graph Generation

**Shui Jie**
School of Computer Science
Peking University
2401112104

**Aleksei Lobanov**
School of Computer Science
Peking University
2401213370

**Lawrence Leroy Chieng Tze Yao**
School of Computer Science
Peking University
2401213369

**Charles Young**
School of Computer Science
Peking University
2401112106

## Abstract

Scene graph generation (SGG) is the task of detecting object pairs and their relations in a visual medium, widely used for captioning, generation, and visual question answering. 2D scene graph generation is a subtask that focuses on generating a 2D graph given an image. While the development of models capable of performing 2D SGG has improved in both accuracy and speed, the computational complexity of the problem and the inherently long-tailed distribution of large, available datasets have led to generation speed and accuracy less than ideal for real-time use. Mainstream approaches focus on two-stage generation, where object detection is performed first, followed by a series of comparisons for relation inference. However, these have an inherent drawback where the computational complexity of detecting $n$ objects and their relationships is $n^2$. More recent models have utilized encoder-decoder structures to reduce generation into a 1-stage problem. Unfortunately, the computation required for these architectures is still too high. Additionally, the model bias caused by long-tailed data distributions remains a key problem in both approaches. In this work, we propose YOLO-SG, a novel SGG framework capable of operating in real-time by decoupling object detection and relation detection and by performing relationship inference with multiple detection models in parallel. Our proposal seeks to both alleviate the effects of the long-tailed distribution problem and perform high-speed inference. Preliminary experiments on the Visual Genome 1.2 dataset demonstrate that YOLO-SG can achieve competitive performance with state-of-the-art models while maintaining high inference speed.

## 1 Introduction

In recent decades, object detection problems have become an increasingly popular subject in research literature[33] as the rise of deep learning[13] has led to many breakthroughs in the field. In this work, we leverage this rapid advancement in object detection to improve upon a downstream task: scene graph generation (SGG)[10]. SGG encompasses the set of tasks that focus on detecting object pairs and their relationships in visual media, creating scene graphs that can then be used for applications such as image captioning[7, 6], image generation[26], and visual question answering[32]. Two significant barriers stand in the way of higher performances for SGG tasks: The long-tailed distribution problem [15, 3] and the quadratic increase in candidate triplets [15, 3, 30]. When collecting data regarding a subject in an image and its relation to other objects, a small number of predicate terms often appear significantly more than all other terms. For example, 'on' can be used to

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

describe the relation between almost any subject-object pair where the subject is 'above' the object, and hence will appear significantly more than predicates such as 'mounted' or 'reaching'. Despite many attempts to alleviate the effect this uneven distribution has on the final results[14, 5, 18, 8], the prediction bias that arises from the uneven data distribution remains an important problem to be solved. The second barrier is the quadratic increase in computation required proportional to the number of detected objects. Mainstream approaches first obtain a list of objects in an image, and then infer the predicate (if one exists) for every possible subject-object pair [30, 10]. Unfortunately, the number of parameters required in the second stage scales quadratically with the number of objects found, as $n$ objects would have $n^2$ potential predicates. Recent works have proposed a one-stage approach that leverages transformers to generate a set of triplets using object information[3, 16]. However, the large number of parameters required in a transformer architecture also slows down the model inference times, preventing the use of these approaches in real-time applications. This paper seeks to address the aforementioned long-tailed data distribution by leveraging the recent advancements in object detection accuracy. To do this, we first reformulate the SGG task into an object detection and classification problem, where each relationship triplet is represented as three objects: the subject, the object, and the relationship predicate. By formulating the problem in this way, we can divide the training data into balanced subsets, where every subset consists of predicate classes with a similar number of instances. This allows us to train multiple detection models, each specialized in detecting a subset of predicates. More importantly, we propose this approach as a solution to the long-tailed distribution problem, as these models will have significantly less bias than a single model trained on all predicate instances. Specifically, we propose a new framework, YOLO-SG, that leverages the YOLOv11 object detection model[21] to detect objects and predicates in parallel. We evaluate our model on the Visual Genome 1.2 dataset[12] and demonstrate that our model can achieve competitive performance with state-of-the-art models while maintaining real-time inference speeds.

## 2 Team Members

**Shui Jie**

As the overall project manager of the team, I am responsible for the overall work of the projects, including different iterations of goals and work distribution. I am responsible for designing the overall pipeline, training the YOLO model, and data analysis on individual model performance on different sets of objects and relationships. I also took part in coding the APIs to integrate my models into the complete pipeline.

**Aleksei Lobanov**

My name is Aleksei Lobanov, I am 23 years old, and I did my undergrad at Shenzhen MSU-BIT University, studying Applied Mathematics, where I did research on Formal Languages. At PKU, I study at the School of CS, NEEC Lab under Professor Luo Guojie. My research interests include programming languages, compiler development, and other systems programming-related topics.

This project was the first time I had to do anything ML-related, it was a big learning experience for me. I had two big tasks: pre-processing the dataset to try to improve the class imbalance within the predicates, and developing a small neural network that tries to assemble the final output of object-predicate-subject triplets. I found working on the neural network particularly challenging, since I had to learn everything from scratch, and I have to thank my amazing teammates for helping me throughout the semester.

**Lawrence Leroy Chieng Tze Yao**

**Charles Young**

My name's Charles Young, I'm a 23-year-old who went to undergrad at UC San Diego for a bachelor's in Mathematics and Computer Science. I study under Professor Tao Xie, and my research interests include mutation testing, metaheuristics, and developer tool use. I was responsible for writing the report and researching related works to improve our framework. I was able to learn a lot about

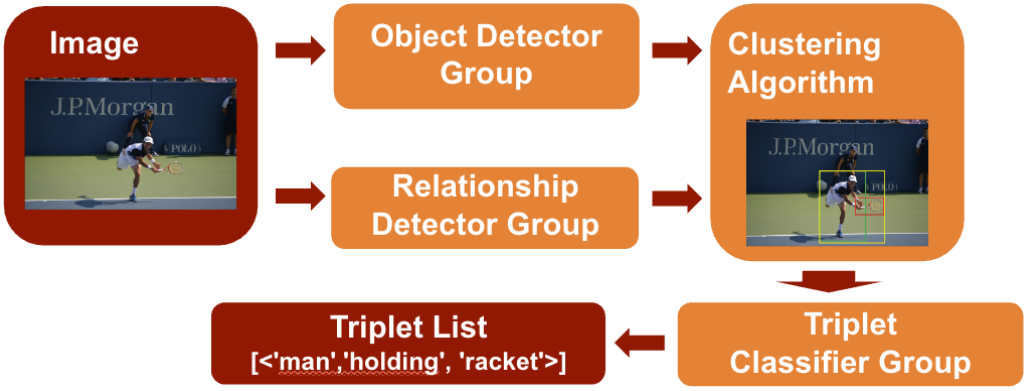


Figure 1: Simple illustration of YOLO-SG's framework.

the current state of the art in scene graph generation and object detection, and I'm grateful to my teammates for their hard work and dedication to the project.

## 3 Background and Related Work

### Scene Graph Generation

Since beginning in 2015 with Johnson et als. work[10], many works have been published improving upon and expanding the scope of SGG[3, 4, 30, 17, 11, 16, 20, 9, 14, 5, 8, 26, 18, 24, 2, 29]. Mainstream works often focus on the development of a two-stage scene graph generation approach composed of an object detection stage and a contextual reasoning stage [30, 11]. This two-stage approach the is foundation for most modern SGG methods[30, 4, 17, 5, 8]. The first stage is often completed using pre-trained detection models[1], while contextual reasoning is done using a variety of methods, from convolutional neural networks[30] to transformers[3]. To address the quadratic growth of computation relative to object count common in two-stage models, recent works have proposed several one-stage approaches to the problem[3, 16]. These models leverage transformer architectures to generate a set of triplets using object information and have been shown to be competitive with two-stage models in terms of performance. However, the large number of parameters required in a transformer architecture also slows down the model inference times, preventing the use of these approaches in real-time applications[3, 18].

**Long-tailed Distribution Problem** Several works have been proposed to mitigate the long-tail bias introduced by the data[24, 31, 2, 9]. These works can be split into two categories: approaches that used biased training data paired with extra learning techniques or approaches that attempt to remove bias during training. In the first category, TDE[24] leverages a causal graph to infer the effect of long-tail bias. CogTree[31] builds a cognitive structure that distinguishes course-to-fine relationships based on biased predictions. DLFE[2] uses Dynamic Label Frequency Estimation to recover unbiased probabilities and reduce reporting bias. In the second category, The second category of works often approaches the problem by preprocessing training data to remove bias. PCPl[29] utilizes the correlation between predicate classes to adjust the loss weights. BGNN[17] uses a bi-level data resampling strategy with a confidence-aware bitartite graph to reduce bias. One particular work that heavily motivated our approach is the Context Knowledge Network (CKN) proposed by Jin et al. [9], which only uses the object labels and bounds as an input to a multilayer perceptron with the number of possible predicates as the output. Despite the lack of visual information, the authors were still able to achieve competitive high-speed performance with state-of-the-art models, whilst also reducing model bias. This work motivated our approach with the use of a lightweight MLP similar to CKN, as Jin et al. demonstrated potential in the ability of machine learning models to extract relationship information from non-visual contexts[9]. However, our approach differs as we use more than one model, with each model specializing in obtaining confidence scores for only a subset of predicates.

**You Only Look Once (YOLO)**

YOLO is a popular object detection model that has been widely used in many computer vision tasks due to its ability to perform single-pass, real-time inferencing. The first version of YOLO proposed by Redmon et al. in 2016 allowed for end-to-end training, as well as object bounding box proposal and object classification in a single forward pass[21]. More importantly, the model was able to achieve this at a rate of 45 frames per second, more than enough for real-time applications. Since then, multiple versions of YOLO have been introduced by various different authors, each of which attempt to improve upon the original whilst following its core design philosophy: open source, end-to-end, and one-shot[25]. These newer versions often apply novel techniques to achieve greater overall performance, measured by inference speed, training cost, and inference accuracy[25, 22, 23], For our framework, we chose to use the YOLOv11 model due to its higher inference speed compared to YOLOv8 due to the lower number of trained parameters[27]. However, it is still possible that performance could be improved using an older version, but this will be left as potential future work.

## 4 Framework Design

YOLO-SG consists of four main components: an object detection group, a predicate detection group, a clustering stage, and a triplet matching group. The object and predicate detection groups are composed of two sets of YOLOv11 models[21]. The first set is trained to detect and label objects, while the second set is trained to detect and label predicates. We then split the training data such that every model is provided with a data points consisting only of a subset of predicates that form a balanced dataset relative to the original dataset. By doing so, we obtain multiple unbiased relationship detection models that bound and label their own set of predicate classes instead of a single heavily biased model that predicts all predicate classes at once. The predicate inference module takes the output of the first stage and obtains a list of objects for each predicate using a simple clustering algorithm. It then passes every possible pair of objects in this list through a lightweight multilayer perceptron which is trained on the same subset of predicate classes. This then returns confidence scores for that subset of predicates. We then return the triplet with the highest confidence score of the detected predicate as the final output. Since we only use the object labels and positions as inputs as opposed to visual features, we can significantly reduce the number of trained parameters compared to methods that rely on convolutional layers or transformer architectures.

**Example** Suppose we have an image containing a man holding a tennis racket. The 'man' and 'racket' would be bounded and labelled by the object detection group, while the 'holding' relationship would be bounded and labelled by the predicate detection group. The clustering stage would then find all objects, including the man and the racket, that are within the bounds of the 'holding' predicate. The triplet matching group would then pass every possible pair of objects in this list through a lightweight multilayer perceptron trained on the 'holding' predicate class and several other classes. The triplet with the highest confidence score for the 'holding' predicate would then be returned as the final output.

$$|\text{predicate classes}| = \frac{|\text{triplets of most frequent class}|}{|\text{triplets of least frequent class}|} \leq \text{threshold}$$

**Preprocessing** Since we need the neural network to interpret words and letters meaningfully, we first convert object and and predicate labels into vectors using GLOVE[19], a word-to-vector library commonly used by other works on SGG tasks[14].

To treat predicates as objects, we assign each annotated predicate to a bounding box that encapsulates both the subject and object bounding boxes. For example, given a pair of objects $A, B \in \mathbb{R}^2$ with predicate $r$ in Visual Genome 1.2, we create a new bounding box $R$ that minimally encloses both $A$ and $B$. such that Our dataset are separated into wo In order to represent relationships and objects rather than edges between objects, we must add a corresponding predicate bounding box for each relationship triplet. We define the predicate bounding box $R$ as the minimal bounding box that encapsulates both the subject and object bounding boxes. Given a pair of objects $A, B$ bounding positions $(x_1, y_1, x_2, y_2)$ where $x_1 < x_2, y_1 < y_2$ and a predicate $r$, we define the predicate bounding

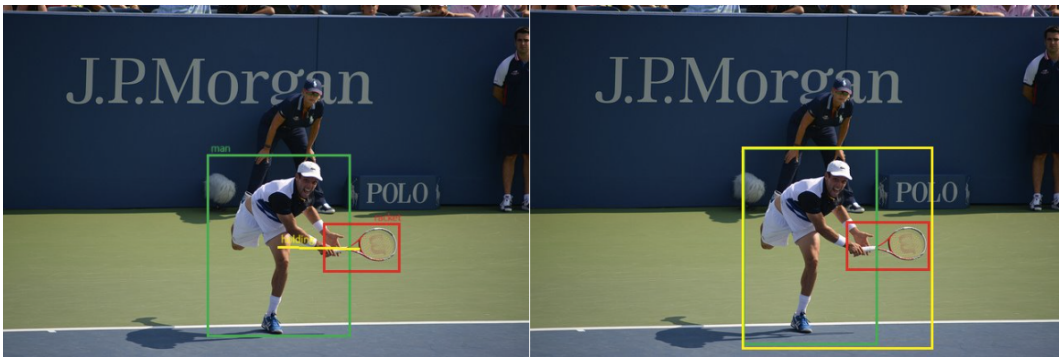


Figure 2: Example of how a predicate bounding box is determined. Given a man holding a racket, the predicate bounding box for 'holding' bounds both the man and the racket.

box $R$ as

$$R = (\min(A[x_1], B[x_1]), \min(A[y_1], B[y_1]), \max(A[x_2], B[x_2], \max(A[y_2], B[y_2]))$$

This ensures that each predicate is represented as a single entity in the image which encompasses the subject and object. For example, given a subject with bounding box $A = [100, 100, 200, 200]$ and an object with box $B = [40, 140, 140, 240]$, the bounds for a predicate would be $R = [40, 100, 200, 240]$. We use the original scene graph annotations to produce a secondary dataset of predicate bounding boxes, each labeled with its corresponding predicate. We now have two datasets which are composed of the same images but different bounding boxes and labels: one has the bounds and labels for objects, while the other has the bounds and labels of predicates.

**Data Splitting** We split the dataset as follows. We first sort the predicates found in the dataset by the number of instances associated with each predicate. Next, we separate a number of predicates into their own subset. We determine how many predicates to place into the set by calculating the ratio between the number of a instances containing the most frequently occuring predicate with the number of instances containing the least frequently occuring predicate. It follows that the larger the resulting number, the more unbalanced the dataset. For every subset, we select the largest number of predicates ordered by frequency such that the ratio does not exceed a chosen threshold.

**Object Detection** Our proposed framework, YOLO-SG, decouples the process of object and predicate detection by framing both as object detection tasks. Rather than inferring predicates between every pair of detected objects in a computationally expensive second stage, we directly detect "predicate entities" in parallel alongside object detection. By modeling predicates as distinct objects, we leverage YOLOv11[27] for both object and predicate recognition.

**Clustering Algorithm** We chose to use an Intersection-Over-Union (IoU) based clustering algorithm to associate detected objects with their corresponding predicates due to its simplicity and low time complexity. Given $n$ objects, the algorithm has a worst-case complexity of $O(n \log n)$. For each predicate entity detected, we calculate the IoU of objects that overlap with the predicate bounds, grouping all objects that have an IOU larger than 0.5 with the predicate bounding box. Due to time constraints, we did not perform experiments to determine the optimal IOU threshold, and this is left as potential future work.

**Triplet Matching** After obtaining a list of objects for each predicate, we need to determine which object pair is most likely to be associated with the predicate. We use a lightweight multilayer perceptron (MLP) to score the plausibility of each object pair. The MLP has a total of <!!!> GLOVE vectors corresponding to the objects, the bou has three fully connected layers with ReLU activation functions. The is trained on the same subset of predicate classes as the predicate detection model. For each predicate detection, we run the MLP once for every potential object pair that can be formed by objects in the list, selecting the triplet which returns the highest confidence score for the predicate class label.

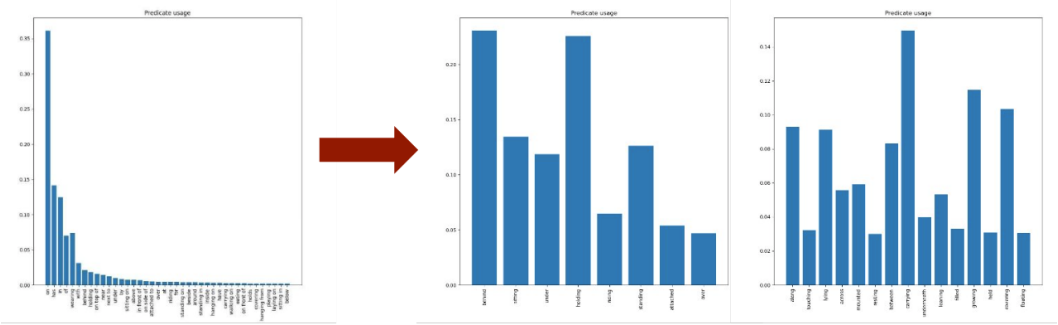

Figure 3: Illustration of dataset splitting results. The left histogram shows the data distribution of the top 50 most frequent predicate classes, while the two right histograms show the distributions of two predicate subsets after splitting. In this example, the threshold ratio is set to 5.

# 5 Experiment

## 5.1 Dataset

We use the Visual Genome (VG) 1.2 dataset, a dataset widely used for SGG benchmarking that is composed of more than 108 thousand images and 2.3 million relationship predicates [12] as our training and test set. note that the Visual Genome dataset contains a large number of object and predicate labels which are semantically equivalent to one another (for example, 'under' and 'below'). In a recent release, the maintainers of the dataset has provided us with a collection of synonym sets that group similar objects and predicates into the same class, with one term assigned by group. After converting all synonymous terms to their respective group representative term, we remove objects which rarely appear along with any associated triplets. After preprocessing, we split VG into training and test sets in the manner proposed by [28] as it is most commonly used by similar works[3, 30]. We end up with a training set containing 73k images and a test set containing 35k images.

## 5.2 Training

Since the data used in every module is independent of the performance of other modules, we train every module in parallel. We train both the object detection and predicate detection models using the YOLOv11 architecture on the same images. However, we only train the object dection models on the bounding box class labels provided by the dataset. For the predicate detection model, we use the predicate bounding boxes that we created during preprocessing. During training, We set the learning rate to 0.0001, the batch size to 32, and the maximum epochs to 300. We train the YOLO models on 5 NVIDIA 4090Ti graphics cards, with training taking 25 hours to complete. The multi-layer perceptrons used for triplet matching are trained using the information of the triplets for the predicate classes that the model is specialized in classifying. Since the number of trained parameters is significantly lower, training the MLPs on a single NVIDIA 3060 graphics card takes only 2 hours.

## 5.3 Evaluation Metrics

The task of scene graph generation can be divided into several subtasks, each with varying levels of difficulty. Predicate classification (PredCLS)[3, 9, 5] predicts predicates given the subject and object labels and bounds. Scene graph classification (SGCLS)[3, 5] predicts the triplet labels given the bounding boxes. Scene graph detection (SGDET)[3], also referred to simply as Scene graph generation (SGGen)[9, 5], predicts the labels and bounding boxes of all three elements. Due to time constraints, we only evaluate the model on the SGGen. This is already sufficient, as SGGen is the most difficult task and the one that we are chiefly concerned about. We follow the common standard set by previous works[3, 30, 16, 9, 29] and adopt the evaluation metric mean recall@K (mR@K). Given a single image, Recall@$k$ is defined as the fraction of ground truth triplets which can be found in the top $k$ triplet predictions. Two triplets are considered equivalent when all three elements (subject, predicate, and object) have been labelled correctly and both object and subject bounding boxes have

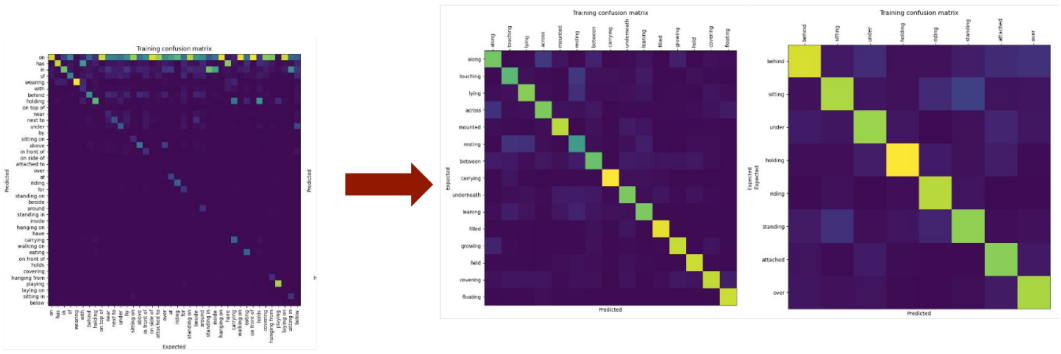

Figure 4: Predicate classification results before and after data splitting. The left image shows the confusion matrix on a sample of predicates for a YOLO model that is given the original dataset in its entirety, while the right matrices demonstrate the results of several models with the same architecture trained on subsets of the sample.

Table 1: Comparison of YOLO-SG with other state-of-the-art models on the Visual Genome 1.2 dataset. the best results in each category are bolded.

| Method | SGGen | | FPS |
|---|---|---|---|
| | mR@50 | mR@100 | |
| RelTr | 10.8 | 12.6 | 16.3 |
| SGTR | 15.8 | **20.1** | 3.8 |
| C-SGG | 14.8 | 17.1 | 33.5 |
| G-RCNN | 6.1 | 7.2 | - |
| BGNN | 10.7 | 12.6 | 2.9 |
| PCPL | 10.4 | 14.4 | - |
| MOTIFS | 5.7 | - | 6.6 |
| **YOLO-SG** (ours) | **17.5** | 17.5 | **43.3** |

IoU > 0.5 compared to the ground truth bounding boxes. We compute the mean recall at K=50 and K=100 for our experiments.

### 5.4 Results

Our model achieves a mean recall of 17.5 at K=50 and K=100, outperforming all other models in the comparison. We also achieve the highest inference speed of 43.3 frames per second, which is significantly higher than other models, clearly demonstrating the effectiveness of our approach. Compared to C-SGG[9], which inspired our MLP triplet matching stage, we achieve both higher recall and higher inference speed. This is likely due to the fact that our model is able to leverage the high accuracy of YOLOv11 in object detection, an important component of the scene graph generation task.

**Threats to Validity** Despite our very promising results, there are many additional changes and further experiments that need to be made in order for the validity of our results to be strong, as there are multiple reasons besides the design of our architecture for the higher performance. Since we used the recently-released YOLOv11 architecture, it is possible that the higher performance is due to the improvements made in the architecture itself, as older models that we compare against use older, less optimized architectures. Additionally, we only evaluate our model on the SGGen task, and it is possible that results on the PredCLS and SGCLS tasks can reveal flaws or further potential improvements in the design of YOLO-SG. Also note that for all other models that we compare YOLO-SG, recall consistently increases with a higher $K$, yet does not increase for YOLO-SG. We believe that this is due to a flaw in our implementation, as the number of predictions must be equal to the number of relationship predicates. Hence, while other models will make multiple predictions with the same predicate but multiple different object-subjects, our will only make one, hence limiting the effect that increasing K has on performance. We can alleviate this problem by including multiple predictions for the same predicate, ranked by confidence score.

## 6 Conclusions

In this work, we propose YOLO-SG, a novel scene graph generation framework that redefines the task of SGG by treating relationships as objects. By using multiple detection models in parallel, We to reduce the long-tail bias caused by the data distribution while at the same time achieving competitive performance and speed with state-of-the-art models. However, there is still a signficant amount of future work required to solidify the validity of the study. We still need to conduct an ablation study to determine the impacts of YOLOv11 on the model's performance by changing to older versions. We also need to conduct exploratory experiments to fine-tune the model parameters, notably the IoU threshold for clustering and the split factor. Finally, we need to resolve some minor flaws that were discovered after evaluation, such as the limiting of predictions to one per predicate found. We believe that with these changes, YOLO-SG can become a powerful tool that offers a new perspective on scene graph generation.

## References


[1] CARION, N., MASSA, F., SYNNAEVE, G., USUNIER, N., KIRILLOV, A., AND ZAGORUYKO, S. End-to-end object detection with transformers. In *European conference on computer vision* (2020), Springer, pp. 213–229.

[2] CHIOU, M.-J., DING, H., YAN, H., WANG, C., ZIMMERMANN, R., AND FENG, J. Recovering the unbiased scene graphs from the biased ones. In *Proceedings of the 29th ACM International Conference on Multimedia* (2021), pp. 1581–1590.

[3] CONG, Y., YANG, M. Y., AND ROSENHAHN, B. Reltr: Relation transformer for scene graph generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 45*, 9 (2023), 11169–11183.

[4] DESAI, A., WU, T.-Y., TRIPATHI, S., AND VASCONCELOS, N. Learning of visual relations: The devil is in the tails. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 15404–15413.

[5] DORNADULA, A., NARCOMEY, A., KRISHNA, R., BERNSTEIN, M., AND LI, F.-F. Visual relationships as functions: Enabling few-shot scene graph prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (2019), pp. 0–0.

[6] GAO, L., WANG, B., AND WANG, W. Image captioning with scene-graph based semantic concepts. In *Proceedings of the 2018 10th international conference on machine learning and computing* (2018), pp. 225–229.

[7] GU, J., JOTY, S., CAI, J., ZHAO, H., YANG, X., AND WANG, G. Unpaired image captioning via scene graph alignments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 10323–10332.

[8] GU, J., ZHAO, H., LIN, Z., LI, S., CAI, J., AND LING, M. Scene graph generation with external knowledge and image reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2019), pp. 1969–1978.

[9] JIN, T., GUO, F., MENG, Q., ZHU, S., XI, X., WANG, W., MU, Z., AND SONG, W. Fast contextual scene graph generation with unbiased context augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 6302–6311.

[10] JOHNSON, J., KRISHNA, R., STARK, M., LI, L.-J., SHAMMA, D., BERNSTEIN, M., AND FEI-FEI, L. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 3668–3678.

[11] JUNG, D., KIM, S., KIM, W. H., AND CHO, M. Devil's on the edges: Selective quad attention for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 18664–18674.

[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* (2017), 32–73.

[13] LECUN, Y., BENGIO, Y., AND HINTON, G. Deep learning. *nature 521*, 7553 (2015), 436–444.

[14] LEE, C.-W., FANG, W., YEH, C.-K., AND WANG, Y.-C. F. Multi-label zero-shot learning with structured knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 1576–1585.

[15] LI, H., ZHU, G., ZHANG, L., JIANG, Y., DANG, Y., HOU, H., SHEN, P., ZHAO, X., SHAH, S. A. A., AND BENNAMOUN, M. Scene graph generation: A comprehensive survey. *Neurocomputing 566* (2024), 127052.

[16] LI, R., ZHANG, S., AND HE, X. Sgtr: End-to-end scene graph generation with transformer. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2022), pp. 19486–19496.

[17] LI, R., ZHANG, S., WAN, B., AND 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* (2021), pp. 11109–11119.

[18] LIANG, X., LEE, L., AND XING, E. P. Deep variation-structured reinforcement learning for visual relationship and attribute detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 848–857.

[19] PENNINGTON, J., SOCHER, R., AND MANNING, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1532–1543.

[20] PLUMMER, B. A., SHIH, K. J., LI, Y., XU, K., LAZEBNIK, S., SCLAROFF, S., AND SAENKO, K. Revisiting image-language networks for open-ended phrase detection. *IEEE transactions on pattern analysis and machine intelligence 44*, 4 (2020), 2155–2167.

[21] REDMON, J. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016).

[22] REDMON, J., AND FARHADI, A. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 7263–7271.

[23] SOHAN, M., SAI RAM, T., REDDY, R., AND VENKATA, C. A review on yolov8 and its advancements. In *International Conference on Data Intelligence and Cognitive Informatics* (2024), Springer, pp. 529–545.

[24] TANG, K., NIU, Y., HUANG, J., SHI, J., AND ZHANG, H. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2020), pp. 3716–3725.

[25] TERVEN, J., CÓRDOVA-ESPARZA, D.-M., AND ROMERO-GONZÁLEZ, J.-A. A comprehensive review of yolo architectures in computer vision: From yolov1 to yolov8 and yolo-nas. *Machine Learning and Knowledge Extraction 5*, 4 (2023), 1680–1716.

[26] TRIPATHI, S., BHIWANDIWALLA, A., BASTIDAS, A., AND TANG, H. Using scene graph context to improve image generation. *arXiv preprint arXiv:1901.03762* (2019).

[27] ULTRALYTICS. Ultralytics yolo11 open-sourced, 2024.

[28] XU, D., ZHU, Y., CHOY, C. B., AND FEI-FEI, L. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 5410–5419.

[29] YAN, S., SHEN, C., JIN, Z., HUANG, J., JIANG, R., CHEN, Y., AND HUA, X.-S. Pcpl: Predicate-correlation perception learning for unbiased scene graph generation. In *Proceedings of the 28th ACM international conference on multimedia* (2020), pp. 265–273.

[30] YANG, J., LU, J., LEE, S., BATRA, D., AND PARIKH, D. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 670–685.

[31] YU, J., CHAI, Y., WANG, Y., HU, Y., AND WU, Q. Cogtree: Cognition tree loss for unbiased scene graph generation. *arXiv preprint arXiv:2009.07526* (2020).

[32] ZHANG, C., CHAO, W.-L., AND XUAN, D. An empirical study on leveraging scene graphs for visual question answering. *arXiv preprint arXiv:1907.12133* (2019).

[33] ZOU, Z., CHEN, K., SHI, Z., GUO, Y., AND YE, J. Object detection in 20 years: A survey. *Proceedings of the IEEE 111*, 3 (2023), 257–276.