
Adaptive Visual Scene Understanding: Incremental Scene Graph Generation

Naitik Khandelwal^{1, 2}, Xiao Liu^{1, 2}, Mengmi Zhang^{1, 2}

¹ College of Computing and Data Science, Nanyang Technological University (NTU), Singapore

² Deep NeuroCognition Lab, Agency for Science, Technology and Research (A*STAR), Singapore
Address correspondence to mengmi.zhang@ntu.edu.sg

Abstract

Scene graph generation (SGG) analyzes images to extract meaningful information about objects and their relationships. In the dynamic visual world, it is crucial for AI systems to continuously detect new objects and establish their relationships with existing ones. Recently, numerous studies have focused on continual learning within the domains of object detection and image recognition. However, a limited amount of research focuses on a more challenging continual learning problem in SGG. This increased difficulty arises from the intricate interactions and dynamic relationships among objects, and their associated contexts. Thus, in continual learning, SGG models are often required to expand, modify, retain, and reason scene graphs within the process of adaptive visual scene understanding. To systematically explore Continual Scene Graph Generation (CSEGG), we present a comprehensive benchmark comprising three learning regimes: relationship incremental, scene incremental, and relationship generalization. Moreover, we introduce a “Replays via Analysis by Synthesis” method named RAS. This approach leverages the scene graphs, decomposes and re-composes them to represent different scenes, and replays the synthesized scenes based on these compositional scene graphs. The replayed synthesized scenes act as a means to practice and refine proficiency in SGG in known and unknown environments. Our experimental results not only highlight the challenges of directly combining existing continual learning methods with SGG backbones but also demonstrate the effectiveness of our proposed approach, enhancing CSEGG efficiency while simultaneously preserving privacy and memory usage. All data and source code are publicly available here.

1 Introduction

Scene graph generation (SGG) aims to extract object entities and their relationships in a scene. The resulting scene graph, carrying semantic scene structures, can be used for a variety of downstream tasks such as object detection[64], image captioning [20, 1], and visual question answering [17]. Despite the notable advancements in SGG, current works have largely overlooked the critical aspect of continual learning. In the dynamic visual world, new objects and relationships are introduced incrementally, posing challenges for SGG models to account for new changes without forgetting previously acquired knowledge. This problem of Continual Scene Graph Generation (CSEGG) holds great potential for various applications, such as real-time robotic navigation in dynamic environments and adaptive augmented reality experiences.

The field of continual learning has witnessed significant growth in recent years, with a major focus on tasks such as image classification [43], object detection [67], and visual question answering [28]. However, these endeavors have largely neglected the distinctive complexities associated with CSEGG. Here, we highlight several unique challenges of CSEGG: (1) In contrast to object detection,

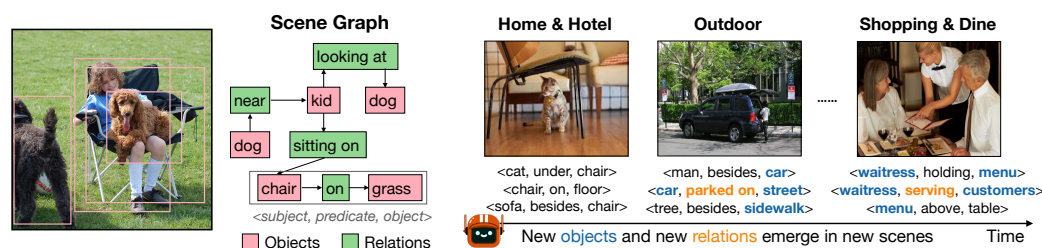


Figure 1: (a) **A scene graph is a graph structure**, where objects are represented as nodes (red boxes), and the relationships between objects are represented as edges connecting the corresponding nodes (green boxes). Each node in the graph contains information such as the object's class label, and spatial location. The edges in the graph indicate the relationships between objects, often described by predicates. A scene graph can be parsed into a set of triplets, consisting of three components: a subject, a relationship predicate, and an object that serves as the target or object of the relationship. The graph allows for a compact and structured representation of the objects and their relationships within a visual scene. (b) **An example CSEGG application** is presented, where a robot continuously encounters new objects (blue) and new relationships (yellow) over time across new scenes.

SGG involves understanding and capturing the relationships between objects, which can be intricate and diverse. Consequently, in CSEGG, conveying the spatial and semantic relationships between objects demands adaptive reasoning from the dynamic scene. (2) SGG introduces a higher level of combinatorial complexity than object detection and image classification because each detected object pair may have multiple potential spatial and functional relationships. Thus, as new objects are introduced to the scenes, the complexity of relationships among all the objects increases significantly in a non-linear fashion. (3) The long-tailed distribution in both objects and relationships in SGG can be attributed to the inherent characteristics of real-world scenes, where certain objects are more prevalent than others. Consequently, CSEGG requires the computational models to adapt continually to the evolving long-tailed distributions over different scenes. Due to a scarcity of research specifically addressing these challenges of CSEGG, there is a pressing need for specialized investigations and methodologies to enable computational models with the ability of CSEGG.

In this study, we re-organize existing SGG datasets [25, 27] to establish a novel and comprehensive CSEGG benchmark with 3 learning protocols as shown in **Fig. 2. (S1)**. Relationship-incremental setting: an SGG agent learns to recognize new relationships among familiar objects within the same scene. **(S2)**. Scene-incremental setting: an SGG agent is deployed in new scenes where it has to jointly learn to detect new objects and classify new relationships. **(S3)**. Relationship generalization setting: an SGG agent generalizes to recognize known relationships among unknown objects, as the agent learns to recognize new objects.

We curate a set of competitive CSEGG baselines by directly combining three major categories of continual learning methods with two SGG backbones and benchmark them in our CSEGG dataset. Their inferior performances show the difficulties of our benchmark tasks, which require the ability to expand, modify, retain, and reason scene graphs within the process of adaptive visual scene understanding. Specifically, the weight-regularization methods fail to estimate the importance of learnable parameters given the complicated model design in SGG backbones. Although image-replay methods retain knowledge from prior tasks through replays, the extensive combinatorial complexity of relationships among objects surpasses the complexity accommodated by a restricted set of replay images with efficient storage. Additionally, none of these baseline methods consider the shifts inherent in long-tailed distributions in dynamic scenes.

To address the CSEGG challenges, we present a method called "Replays via Analysis by Synthesis", abbreviated as RAS. RAS employs scene graphs from previous tasks, breaks them down and re-composes them to generate diverse scene structures. These compositional scene graphs are then used for synthesizing scene images for replays. Due to its nature of symbolic replays, RAS does not require the storage of original images, which often carry excessive and redundant details. This also ensures data privacy preservation and data efficiency. Furthermore, by synthesizing scenes using composable scene graphs, RAS maintains the semantic context and structure of previous scenes and also enhances the diversity of scene generation. To prevent biased predictions stemming from long-tailed distributions, we moderate the distribution of replayed scene graphs by balancing tail and head classes. This ensures a uniform sampling of relationships and objects during replays. Extensive

S.	#Tasks	#Objs	#Rels	Eval. metrics	SGG Backbone	CL base.	Kn.	Unk.
S1	5	150 (<i>All</i>)	10 <i>per task</i>	F, R, mF, mR, FWT, BWT, Gen R _{bbox} , Gen R	Transformer based (SGTR)	Joint, Naive, Replay M%, EWC, PackNet, RAS_GT	Objs (bbox, labels)	Rels
S2	2	<i>Task 1</i> : 100 <i>Task 2</i> : 25	<i>Task 1</i> : 40 <i>Task 2</i> : 5				None	Rels and Objs (bbox, labels)
S3	4	30 <i>per task</i>	35 <i>per task</i>		CNN based (IMP)		Rels	Objs (bbox, labels)

Table 1: **Overview of three CSEGG learning scenarios.** This table summarizes the three learning scenarios (Column 1) in CSEGG, including the number of tasks, the number of object (#Objs) and relationship (#Rels) classes, the evaluation metrics, the SGG-Backbones used, and the continual learning (CL) baselines. The Kn. and Unk. columns provide information regarding what is known to the CSEGG models during training in that scenario and what is being incrementally learned by the models. Unknown information is being incrementally learned by the models. See **Sec. 3** for details.

experiments underscore the effectiveness of our approach. Network analysis reveals our crucial design choices that can be beneficial for the future development of CSEGG models.

2 Related Works

Scene Graph Generation Datasets. Visual Phrase [59] stands as one of the earliest datasets in the field of visual phrase recognition and detection. Over time, various large-scale datasets have emerged to tackle the challenges of Scene Graph Generation (SGG) on static images [23, 42, 25, 27, 37, 78, 74, 72, 80, 12, 35, 81]. Subsequent works further extend the SGG to dynamic videos [22, 50, 56]. Despite the significant contributions of these datasets to SGG, none focuses on continual learning in SGG. As the preliminary efforts towards CSEGG, we start with fundamental and straightforward settings of SGG on static images. Among all the SGG datasets on static images, the Visual Genome dataset [25] has played a pioneering role by providing rich annotations of objects, attributes, and relationships in images. Thus, we re-structure the Visual Genome dataset [25] and establish a novel and comprehensive CSEGG benchmark, where AI models are deployed to dynamic scenes where new objects and new relationships are introduced.

Scene Graph Generation (SGG) Models. SGG models are categorized into two main approaches: top-down and bottom-up. Top-down approaches[38, 77] typically rely on object detection as a precursor to relationship prediction. They involve detecting objects and then explicitly modeling their relationships using techniques such as rule-based reasoning[42] or graph convolutional networks [73]. On the other hand, bottom-up approaches focus on jointly predicting objects and their relationships in an end-to-end manner [34, 35, 72]. These methods often employ graph neural networks [33, 82] or message-passing algorithms [72] to capture the contextual information and dependencies between objects. Furthermore, recent works have explored the integration of language priors [48, 42, 69] and attention mechanisms in transformers [3] to enhance the accuracy and interpretability of scene graph generation. However, none of these works evaluate SGG models in the context of continual learning. In our work, we directly combine continual learning methods with SGG backbones and benchmark these competitive baselines in CSEGG. Our results reveal the limitations of these methods and highlight the challenges of our CSEGG learning protocols.

Continual Learning Methods. Existing continual learning works can be categorized into several approaches. (1) Regularization-based methods [24, 9, 79, 2, 4] aim to mitigate catastrophic forgetting by employing regularization techniques in the parameter space. (2) Dynamic architecture-based approaches[66, 76, 21, 47] adapt the model’s architecture dynamically to accommodate new tasks without interfering with the existing ones. (3) Replay-based methods [57, 10, 55, 65, 52, 7] utilize a memory buffer to store and replay past data during training, enabling the model to revisit and learn from previously seen examples, thereby reducing forgetting. The special variants of these methods include generative replay methods, such as [61, 71, 75, 51], where synthetic data is generated and replayed. Although these generative replay methods, as well as other continual learning methods, have been extensively studied in image classification [8, 70, 43] and object detection[68, 60, 45], few works focus on the challenges in CSEGG, such as adaptive reasoning from the dynamic scenes, the evolving long-tailed distribution across scenes, and the combinatorial complexity involving objects and their multiple relationships. In this work, we introduce a continual learning method, abbreviated as RAS (Replays via Analysis by Synthesis). To address the distinct challenges of CSEGG, RAS

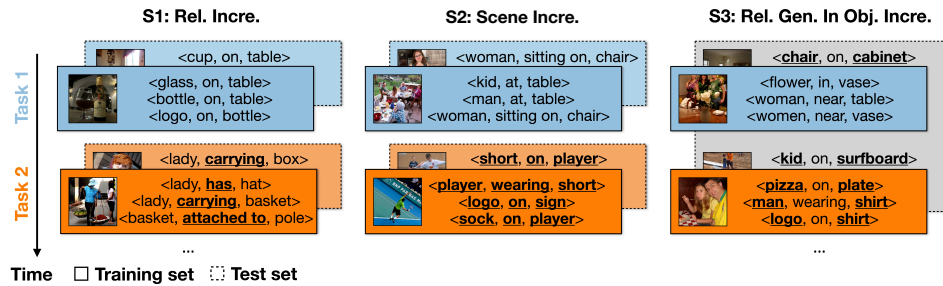


Figure 2: **Three learning scenarios** are introduced. From left to right, they are S1. relationship (Rel.) incremental learning (Incre.); S2. scene incremental learning; and S3. relationship generalization (Rel. Gen.) in Object Incre.. In S1 and S2, example triplet labels in the training (solid line) and test sets (dotted line) from each task are presented. The training and test sets from the same task are color-coded. Blue color indicates task 1 and orange color indicates task 2. The new objects or relationships in each task are bold and underlined. In S3, one single test set (dotted gray box) is used for benchmarking the relationship generalization of object incre. learning models across all the tasks.

involves creating in-context synthetic scene images based on re-composable scene graphs from previous tasks to reinforce continual learning. The components in RAS facilitate memory-efficient training and preserve privacy while maintaining the scene diversity and scene context for SGG in dynamic environments. With the rise of pretrained vision-language models (VLMs) [49, 31, 83, 40], various SGG methods [11, 30] have been proposed to tackle open-vocabulary and zero-shot SGG challenges. However, these settings differ fundamentally from CSEGG, where we aim to simulate scenarios where the model encounters novel predicates or objects, unseen by any models, including LLMs or multi-modal models. Using pre-learned information from LLMs or frozen encoders conflicts with the continual learning setting we address in CSEGG.

3 Continual Scene Graph Generation Benchmark

In CSEGG, to cater to the three continual learning scenarios below, we re-organize the Visual Genome [25] dataset and follow its standard image splits for training, validation, and test sets specified in [72]. In each learning scenario, we consider a sequence of T tasks consisting of images and corresponding scene graphs with new objects, or new relationships, or both. Let $D_t = \{(I_i, G_i)\}_{i=1}^{N_t}$ represent the dataset at task t , where I_i denotes the i -th image and G_i represents the associated scene graph. The scene graph G_i comprises a set of object nodes O_i and their corresponding relationships R_i . Each object node o_j is defined by its class label c_j and its bounding box locations and sizes b_j . Each relationship r_k is represented by a triplet (o_s, p_k, o_o) , where o_s and o_o denote the subject and object nodes, and p_k represents the relationship predicate.

3.1 Learning Scenarios

Scenario 1 (S1): Relationship Incremental Learning. To uncover contextual information and go beyond studies of object detection and recognition, we introduce this scenario consisting of 5 tasks where 10 new relationship classes are incrementally added in every task (**Fig. 2, left; Fig. S1; Tab. 1**). All object classes and their locations are made known to all CSEGG models over all the tasks. This scenario resembles a human learning scenario where a parent gradually teaches a baby to recognize new relationships among all objects in the same room, focusing on one new relationship at a time during continual learning. This scenario also has implications in medical imaging where identical cell types may form new relationships with nearby cells depending on the context (**Sec. A.1.1**).

Scenario 2 (S2): Scene Incremental Learning. To simulate the real-world cases when there are demands for detecting new objects and new relationships from old to new scenes, we introduce this scenario where new objects and new relationships are incrementally introduced over tasks (**Fig. 2, middle; Fig. S1; Tab. 1**). There are 2 tasks in total with the first task containing 100 object classes and 40 relationship classes with 25 more object classes and 5 more relationship classes in the second task. This aligns with the real-world use cases where common objects and relationships are learned in the first scene, and incremental learning in the second scene only happens on less frequent relationships and objects. See **Sec. A.1.2** for details.

Scenario 3 (S3): Relationship Generalization. Humans have no problem at all recognizing the relationships of unknown objects with other nearby objects. This scenario is designed to investigate the relationship generalization ability of CSEGG models. This capability is essential for real-world implications, such as in robotic navigation where it often encounters unknown objects and requires classifying their relationships. In total, there are four tasks, each introducing an incremental addition of 30 new object classes. All relationship classes are made known to all CSEGG models over all the tasks (**Fig. 2, right; Fig. S1; Tab. 1**). Different from scenarios **S1** and **S2**, a standalone generalization test set is curated, where the objects are unknown but the relationship classes among these unknown objects are common to the training set of every task. The CSEGG models trained after every task are tested on this standalone generalization test set to predict relationships among the unknown objects. See **Sec. A.1.3** for details.

Data sampling and distributions. To allocate data for every task of each scenario, we perform the following sampling strategies. In **S1** and **S3** above, either object or relationship classes are randomly sampled from the Visual Genome dataset and incrementally added to every task. Due to the inherent characteristics of real-world scenes, the long-tailed class distribution is present in **S1** and **S3**. However, in **S2**, only tail classes are sampled and added in subsequent tasks. The number of tasks in each scenario is experimentally determined to optimize the training data configuration, ensuring sufficient training samples in each task while maximizing the number of tasks. For detailed statistics, see **Fig. S2** and **Sec. A.2**.

3.2 Competitive CSEGG Baselines

Due to the scarcity of CSEGG works, we contribute a diverse set of competitive CSEGG baselines and implement them on our own. Each CSEGG baseline requires three components: a backbone model for scene graph generation (SGG), a continual learning (CL) method to prevent the SGG model from forgetting, and an optional data sampling technique to deal with imbalanced data at every task for training SGG models. Next, we introduce the 2 SGG backbones, the 5 continual learning methods, and the 5 optional data sampling techniques. See **Sec. A.3** for implementation and training details of CSEGG baselines.

SGG Backbones. We use the two state-of-the-art backbones: (1) one-stage Scene graph Generation TRansformer (SGTR) [32] and (2) the traditional Two-stage SGG model (TCNN) [72]. Briefly, SGTR (**Fig. S3 left**) uses a transformer-based architecture for image feature extraction and fusion. During training, [32] formulates SGG as a bipartite graph construction and matching problem. In contrast, TCNN detects objects with Faster-RCNN[18] backbone and infers their relationships separately via Iterative message passing [72]. We use implementations from [32] and [68] with default hyperparameters.

Baselines. We include the following continual learning methods (**Fig. S3 right**): (1) Naive (lower bound) is trained on each task in sequence without any measures to prevent catastrophic forgetting. (2) EWC[24] is a weight-regularization method, where the weights of the network are regularized in the parameter space, based on their “importance” to the previous tasks. (3) PackNet[44] is a parameter-isolation method, iteratively pruning the network parameters after every task, so that it can sequentially pack multiple tasks within one network. (4) Replay@M[57] includes a memory buffer with the capacity of storing M percentages of images in the entire dataset as well as their corresponding ground truth object and predicate notations depending on the task at each learning scenario. We vary $M = 10\%$, 20% , and 100% . (5) Joint Training is an upper bound where the SGG model is trained on the entire CSEGG dataset. (6) RAS_GT is a baseline in which we use the ground truth scene graph labels from each task to create replay buffers using an image generation model explained in detail in **Sec. 4**. See **Fig. S4** for schematics of CSEGG baselines. We provide mathematical formulations of these baselines in **Sec. A.4**.

Sampling Methods to Handle Long-Tailed Distribution. We adopt the five data sampling techniques to alleviate the problem of imbalanced data distribution during training. (1) LVIS[19] is an image-level over-sampling strategy for the tailed classes. (2) Bi-level sampling (BLS) [33] balances the trade-off between image-level oversampling for the tailed classes and instance-level under-sampling for the head classes. (3) Equalized Focal Loss (EFL) [29] is an effective loss function, re-balancing the loss contribution of head and tail classes according to their imbalanced distribution. EFL is enabled all the time for all the CSEGG baselines. In addition to applying data sampling techniques to the training sets, we can also apply LVIS and BLS techniques to the data stored in the

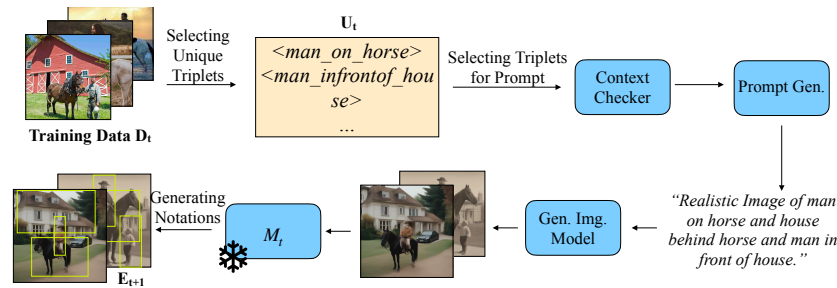


Figure 3: **Schematic of our proposed Replays via Analysis by Synthesis (RAS) method.** At task $t+1$, our RAS stores all the triplet labels U_t , such as $\langle \text{man, on, horse} \rangle$, from the previous tasks. It then re-composes these triplet labels to create in-context prompts, utilizing them as inputs to generative image models to synthesize images for replays. For predicting scene graphs on these synthesized images, we employ the frozen model M_t from the preceding task t , marked with “snowflakes”. Subsequently, these predicted scene graph notations, along with their corresponding synthesized images, contribute to “pseudo” replays, preventing the current model M_{t+1} from experiencing forgetting. See Sec. 4 for more details.

replay buffer. We name these data sampling techniques applied during both training and replays as (4) LVIS@Replay and (5) BLS@Replay.

3.3 Evaluation Metrics

Same as existing SGG works [72, 32], we adopt the evaluation metric recall@K (**R@K**) on the top K predicted triplets in the scene graphs G . As CSEGG is long-tailed, we further report the results in mean recall (**mR@K**) over the head, body, and tail classes. Forgetfulness (F), Average (Avg.) performance, Forward Transfer (FWT) [39] and Backward Transfer (BWT) [41] are standard evaluation metrics used for continual learning in image recognition and object detection tasks. In Scenario 1 and 2, we adapt these metrics to recalls **R@K** and introduce **F@K**, **Avg.R@K**, **FWT@K**, and **BWT@K** respectively for CSEGG settings. Similarly, we also adapt these metrics to **mR@K**. We explored CSEGG with $K=20, 50$, and 100 . Since our results are consistent among K s, we omit “@K” and analyze all the results based on $K=20$ in the entire text.

In scenario S3, we evaluate all CSEGG methods in the standalone generalization test set, shared over all the tasks. To benchmark generalization abilities in unknown object localization and relationship classification among these unknown objects, we introduce two evaluation metrics: **Gen R_{bbox}@K** and **Gen R@K**. As the CSEGG models have never been taught to classify unknown objects, we discard the class labels of the bounding boxes and only evaluate the predicted bounding box locations with **Gen R_{bbox}@K**. To evaluate whether the predicted bounding box location is correct, we apply a hard threshold of Intersection over Union (**IoU**) between the predicted bounding box locations and the ground truth. Any predicted bounding boxes with their IoU values above the hard threshold are deemed to be correct. We vary IoU thresholds from 0.3, 0.5, to 0.7.

To assess whether the CSEGG model generalizes to detect known relationships over unknown objects, we evaluate the recall **Gen R@K** of the predicted relationships r_k only on *correctly predicted* bounding boxes. See Sec. A.5 for details. All results are averaged over 3 runs.

4 Replays via Analysis by Synthesis (RAS)

To address the complexities in CSEGG, we introduce our “Replays via Analysis by Synthesis” method, dubbed RAS. Our RAS belongs to the group of generative replay methods for continual learning. We create an exemplar set E_t for replays. At task t , we jointly train the scene graph generation model M_t on E_t and the current training dataset D_t . However, different from the existing generative replay methods [61, 15, 16], our RAS leverages symbolic replays with state-of-the-art diffusion models. Moreover, rather than generating any random images for replays, our RAS is capable of generating in-context images following semantic rules, such as object co-occurrences. The schematic of our RAS is presented in Fig. 3. Next, we focus on how RAS creates E_t containing the generated images and their SGG annotations on these images for replays.

Image Generation. At the current task $t + 1$, our RAS requires storing the frozen old model snapshot M_t at the end of the previous task t and all the triplet labels U_t , which are parsed from all the scene graphs aggregated from all the previous tasks. These triplets contain object labels, subject labels, and relationships among them. For example, $\langle \text{man, on, horse} \rangle$ and $\langle \text{man, in front of, horse} \rangle$ are two unique triplet labels. Unlike the traditional replay methods in continual learning literature, our method refrains from storing original images I_i or scene graphs G_i in the training sets, thereby eliminating storage issues and privacy concerns.

To generate images I'_j for replays, our RAS feeds text prompts, which are formed by a set of chosen triplet labels and describe the diverse in-context scenes, into the state-of-the-art Stable Diffusion model [58]. As previous works suggests context plays important roles in visual perceptions [5]. To generate text prompts describing context-congruent scenes, we employ a context checker. First, the context checker uses the pre-trained large language model BERT [14] to extract embeddings for each triplet label in U_t . As BERT has been pre-trained on a large corpus of text data, it learns to capture context-relevant representations of words. Next, hierarchical clustering is performed on these embeddings using the agglomerative clustering algorithm [46]. This ensures that each cluster contains only embeddings that are semantically close. The threshold for the agglomerative clustering algorithm is set to 0.6. As real-world images often contain complex scenes involving multiple triplets, we select any cluster with more than 3 triplet labels to create a text prompt for image generation.

In practical applications, conducting agglomerative clustering on all triplet labels in U_t is computationally demanding, as it requires computing pairwise embedding similarities among all the triplet labels. To address this, RAS opts for a more efficient approach during replays by selecting a subset of triplet labels and clustering their embeddings. Recognizing that real-world scenes often exhibit a long-tailed distribution with certain objects or relationships being more prevalent, we introduce the Long-Tailed Distribution (LTD) module for balancing this distribution in U_t . Unlike image-level sampling methods like BLS and LVIS [33, 19] discussed in **Sec. 3.2**, our LTD module in RAS operates at the triplet level. For each triplet label, its dropout rate is determined proportionally to its frequency in U_t . Specifically, we define the drop-out rate d_k for the k -th triplet as: $d_k = f_k / (\sum_{i=1}^{i=N} f_i) * \alpha$, where N is the total number of triplets in U_t , $\alpha = 0.7$ is a scaling factor, and f_i is the frequency of the i -th triplet in U_t . This sampling formula enables RAS to select triplets from tail classes more frequently compared to those from head classes.

To generate a text prompt from the chosen triplet labels, we employ a straightforward English language construct using the conjunction "and". This involves combining all the selected triplet labels into a sentence by starting with "Realistic Image of". For instance, if the triplet labels are $\langle \text{man, on, horse} \rangle$, $\langle \text{house, behind, horse} \rangle$, and $\langle \text{man, in front of, house} \rangle$, the generated prompt becomes "Realistic Image of man on horse and house behind horse and man in front of house." To increase exemplar diversity for replays, we use the Stable Diffusion model [58] to generate γ number of images for the same text prompt. In practice, we set $\gamma = 10$ over all the learning scenarios.

We provide the visualization examples of some synthesized images along with the corresponding text prompts in **Fig. S5**. From these examples, we found that the composed text prompts and the synthesized images are often of high quality and contextual coherence.

Scene Graph Prediction on Synthesized Images. During replays, to train the model M_{t+1} on I'_j , we also need to predict their corresponding scene graph notations G'_j on I'_j . As the frozen model snapshot M_t at the end of task t carries prior knowledge for SGG from the previous tasks, we use it to predict notations G'_j on I'_j . These G'_j comprises object nodes O'_j with their respective classes c'_j , along with object locations b'_j . Additionally, it includes corresponding relationship nodes R'_j formed by triplets $\langle o'_s, p'_k, o'_j \rangle$ representing subject, predicate, and object nodes, respectively. These generated notations G'_j , along with I'_j , serve to construct the exemplars E_t , used for replays.

5 Results

5.1 RAS outperforms all the CSEGG baselines in Scenarios 1 and 2

The results for Avg. R, F, mR, mF, FWT, and BWT in learning scenarios 1 (S1) and 2 (S2) are presented in **Tab. 2**. Our observations align with established research in continual learning, especially in image classification and object detection: regardless of the SGG architectures, over both learning scenarios, Naive consistently performs the worst, showcasing significant catastrophic forgetting.

SGTR[32]												
Methods	Learning Scenario 1 (S1)						Learning Scenario 2 (S2)					
	Avg.R↑	F↑	mR↑	mF↑	FWT↑	BWT↑	Avg.R↑	F↑	mR↑	mF↑	FWT↑	BWT↑
Joint	20.15	0	4.6	0	-	-	12.64	0	9.84	0	-	-
Replay@100%	16.17	-12.24	3.32	-1.34	-1.77	-11.72	4.56	-4.13	4.56	-5.61	-1.045	-30.25
Naive	1.33	-28.7	0.86	-1.74	-2.03	-60.67	0.51	-23.22	0.05	-11.31	-3.77	-62.34
EWC[24]	1.89	-28.4	0.96	-1.72	-1.17	-52.45	0	-23.22	0	-11.31	-2.65	-50.12
RAS_GT	5.78	-26.51	1.43	-1.54	-1.2	-44.27	0.98	-23.11	0.76	-10.86	-1.6	-43.25
PackNet[44]	7.19	-25.67	1.35	-1.64	-1.03	-42.35	1.67	-22.77	0.9	-10.33	-1.4	-42.45
Replay@10%	8.55	-22.21	4.33	-1.44	4.29	-38.35	1.81	-20.72	1.15	-9.64	-0.9	-40.67
Replay@20%	9.25	-20.35	4.78	-1.42	3.21	-31.98	2.57	-17.17	1.56	-8.07	-0.67	-38.27
Ours*	10.78	-18.92	5.6	-1.39	2.3	-25.56	3.45	-10.23	2.75	-6.57	-0.54	-35.67

TCNN[72]												
Methods	Learning Scenario 1 (S1)						Learning Scenario 2 (S2)					
	Avg.R↑	F↑	mR↑	mF↑	FWT↑	BWT↑	Avg.R↑	F↑	mR↑	mF↑	FWT↑	BWT↑
Joint	19.53	0	3.9	0	-	-	4.3	0	3.7	0	-	-
Replay@100%	13.45	-8.83	3.6	-0.35	-1.5	-10.45	12.45	-4.13	3.2	-0.56	-2.1	-20.34
Naive	0.98	-21.2	0.74	-1.35	-3.45	-43.87	0	-18.22	0.45	-2.67	-4.12	-53.12
EWC[24]	2.36	-21.05	0.67	-1.34	-2.34	-39.89	0	-18.22	0.03	0	-3.77	-51.67
PackNet[44]	3.2	-19.7	1.1	-1.13	-1.3	-32.45	1.1	-17.82	0.84	-1.97	-2.84	-40.34
Replay@10%	5.67	-18.9	3.21	-1.05	1.45	-28.34	1.81	-16.72	1.03	-1.74	-1.4	-43.56
Replay@20%	6.23	-17.45	3.5	-1.01	1.01	-24.32	2.37	-15.17	1.45	-1.53	-1.1	-38.56
Ours*	7.8	-15.67	3.9	-0.95	0.5	-19.83	4.67	-11.31	2.2	-0.89	-0.97	-29.65

Table 2: **Results of CSEGG for various continual learning methods applied on the two SGG backbones (SGTR and TCNN) in Learning Scenarios 1 and 2.** See Sec. 3.2 for continual learning baselines. See Sec. 3.3 for evaluation metrics. The higher the evaluation metrics, the better. The best are in bold. * means the experiment is still running, we will report the results in the final version.

Model	mR↑	mF↑
LVIS@Replay@10%	3.98	-1.54
BLS@Replay@10%	4.34	-1.47
RAS (ours)	5.6	-1.39

Table 3: **Results at Task 5 in Learning Scenario 1 when sampling techniques are applied to long-tailed distribution data.** See Sec. 3.2 for the sampling techniques on long-tailed distributions. We copy the results of our RAS from Tab. 2 for easy comparisons. The best results are in bold.

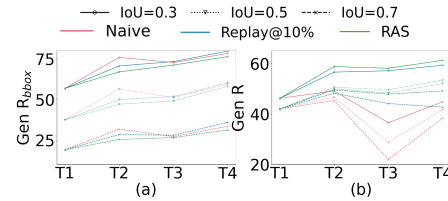


Figure 4: **Results in Scenario 3.** See Sec. 3.3 for evaluation metrics. The higher the values, the better. Line colors indicate continual learning methods. Line types denote the IoU thresholds.

Replay-based methods, such as Replay@10% and Replay@20%, outperform techniques like EWC and PackNet. However, none of them surpass our RAS.

RAS achieves superior performance compared to Replay@20% (~2 Gb) while requiring less storage (~1.2 Gb), equivalent to storing 15% exemplary images. This storage efficiency is due to only needing to store the old model snapshot, triplet labels in U_t , and the image generation model, thus avoiding privacy concerns. Also, RAS outperforms RAS_GT (Tab. 2), indicating that decomposing scene graphs into smaller, more diverse ones, along with more comprehensible prompts, is more effective than storing the ground truth scene graphs and directly using them for image generation.

In Learning Scenario 2 (S2), the task involves classifying both new objects and new relationships, significantly escalating the level of difficulty compared to S1. As evident from Tab. 2, all CSEGG models, including Replay@100%, exhibit a decline in overall performance when compared to the upper bound Joint. Although our RAS achieves the leading performance among all the baselines, its performance is still far from Joint. This underscores the persistent challenge posed by S2 in the context of CSEGG. Future work should explore new approaches to address this gap.

To tackle the issue of imbalanced data distribution in real-world scenarios, we incorporate two established data sampling techniques (LVIS@Replay@10% and BLS@Replay@10%) into our experiment (Sec. 3.2). The outcomes in learning scenario S1 are presented in Tab. 3. We observed that their performance falls short of our RAS, underscoring the efficacy of RAS in addressing long-tailed distribution during generative replays. We also noticed that BLS@Replay@10% significantly

	γ	Context	LTD	Triplet	Avg.R \uparrow	F \uparrow	mR \uparrow	mF \uparrow
A1	10	\times	\checkmark	Multiple	8.23	-21.35	3.98	-1.98
A2	10	\checkmark	\times	Multiple	9.75	-20.65	4.12	-1.65
A3	10	\checkmark	\checkmark	Single	7.75	-22.45	3.12	-2.48
A4	2	\checkmark	\checkmark	Multiple	2.45	-27.43	0.45	-9.84
A5	4	\checkmark	\checkmark	Multiple	5.67	-26.42	2.41	-3.26
A6	8	\checkmark	\checkmark	Multiple	7.89	-22.42	3.89	-2.17
Ours	10	\checkmark	\checkmark	Multiple	10.78	-18.92	5.6	-1.39

Table 4: **Ablation results of our RAS on learning scenario S1 reveals key design insights.** This table presents the results of ablation studies conducted to identify key components of our method, as discussed in **Sec. 5.3**. Results in Avg.R, F, mR, mF are reported after the last task in Scenario S1.

outperforms LVIS@Replay@10%, contrary to findings in the classical SGG problem where BLS is considered more effective than LVIS [33]. The performance difference may stem from variations in the number of replay instances between the two approaches after applying these data re-sampling methods to exemplar images in the memory buffer (**Sec. 3.2**). This observation suggests that the original sampling methods designed for addressing long-tailed distributions in the classical SGG problem may not be as effective when applied to CSEGG.

We explored the impact of task sequence permutations on CSEGG performance, finding an effect consistent with existing literature [63] (**Fig. S6; Sec. A.6.1**). We also observed that fine-tuning DETR in S1 has minimal impact on forgetting, indicating that any forgetfulness in S1 is solely due to relationship incremental learning (**Fig. S7; Sec. A.6.2**). Moreover, to gain qualitative insights, we provide visualizations of predicted scene graphs for all CSEGG baselines in Scenario 1 (**Fig. S8 and Sec. A.7.1**) and Scenario 2 (**Fig. S9 and Sec. A.7.2**).

5.2 CSEGG Models Can Generalize in Unknown Scenes

Fig. 4 illustrates the generalization results for detecting unknown objects and classifying known relationships among these objects in Learning Scenario 3 (S3). In **Fig. 4 (a)**, an increasing trend in Gen R_{bbox} is observed with the increasing task number for all CSEGG methods, indicating improved generalization in detecting unknown objects. Notably, even with minimal training in Task 1, all CSEGG methods propose 23% reasonable object regions with threshold IoU = 0.7, showcasing the SGTR model’s ability to generalize to locate “objectness”. As expected, with an increase in IoU threshold from 0.3 to 0.7, we found that Gen R_{bbox} decreases due to fewer bounding boxes being considered correct. Moreover, we also compared the generalization performance in object detection between Replay@10% and Naive. Contrary to previous observations in S1 and S2 (**Tab. 2**), we found that Replay@10% show a decline in Gen R_{bbox} , possibly due to a fixed number of detected object bounding boxes output by CSEGG methods. Similarly, our RAS exhibits a reduced $G R_{bbox}$ compared to Replay@10% and Naive, likely for the same underlying reason.

In **Fig. 4 (b)**, Replay@10% outperforms Naive in Gen R when considering correctly detected unknown object locations, emphasizing that minimizing forgetting in continual learning enhances the SGTR model’s overall relationship generalization in unknown scene understanding. However, the performance of Replay@10% is still inferior to our RAS method; implying that our RAS is more proficient in generalizing to classify relationships among unknown objects. Interestingly, we also noted that even with minimal training in Task 1, all the CSEGG methods achieve 45% recall of known relationships among unknown objects, demonstrating the SGTR model’s ability to generalize to classify “relationship”. Visualization examples, when CSEGG models can generalize to recognize relationships, are presented in **Fig. S10 and Sec. A.7.3**.

5.3 Ablation Studies on Our RAS Reveal Key Design Insights

We introduce our default method designs in **Sec. 4**. Here, we vary the components in our RAS to reveal key design insights. We propose a context checker in RAS. Here, we conduct an ablation by removing this module. Triplet labels are randomly selected and combined for text prompts. In A1 of **Tab. 4**, we observe a performance decrease of approximately 2% across all evaluation metrics, compared with our RAS. This suggests that generating images adhering to real-world context rules is crucial for replays. The lower performance may be attributed to the challenge of generating

good-quality out-of-context images for Stable Diffusion Models and the potential domain differences affecting the SGG model M_t in predicting out-of-context SGG notations.

The LTD sampling module in our RAS is designed to balance the distribution of head and tail triplet labels from U_t . Here, we remove the LTD sampling module and report the performance of the ablated method in A2 of **Tab. 4**. Compared to our RAS, we observe an absolute decrease of 1-2% across all metrics. Notably, the relative decrease is more pronounced in mR and mF than Avg.R and F. As mR and mF indicate mean Recall and mean Forgetfulness over both tail and head classes, the larger drops in these metrics suggest that the absence of LTD sampling significantly hinders the SGG model's ability to predict tail classes from previous tasks.

In our RAS, we employ multiple triplet labels to construct text prompts for image generation. In contrast to single triplet labels, our approach yields rich text descriptions of complex scenes, allowing the SGG model to capture intricate relationships among multiple objects in the same scene. Additionally, using multiple triplets is more efficient in rehearsing, as it enables the model to practice predicting multiple triplets simultaneously within the same number of synthesized images. Indeed, when we replace multiple triplet labels with single triplet labels for text prompts, we note a decrease of approximately 3% across all metrics (compare A3 with ours in **Tab. 4**).

Lastly, we investigate the impact of generating γ images using the same text prompt in RAS, varying γ from 2 to 8 (**Tab. 4**, A4-6). As expected, performance improves with higher γ , showing that increased sample diversity enhances CSEGG performance. With ample computing resources, dynamically synthesizing more images could further improve performance. This highlights RAS's advantage in generating numerous images for replays without expanding storage usage.

6 Discussion

In the dynamic world, adapting scene graph generation (SGG) models to new objects and relationships poses challenges. Despite progress in SGG and continual learning, there is still a gap in understanding Continual Scene Graph Generation (CSEGG). We address this by operationalizing CSEGG, and introducing benchmarks, datasets, and evaluation protocols. Our study explores three learning scenarios, analyzing continual object detection and relationship classification in long-tailed class-incremental settings for CSEGG baselines. Our findings show that integrating sampling methods with CSEGG baselines to address long-tailed distributions moderately eliminates forgetting; however, a large performance gap between current CSEGG baselines and the joint training upper bound persists. To address CSEGG challenges, we propose RAS, a Replays via Analysis by Synthesis method. RAS parses previous task scene graphs into triplet labels for diverse in-context scene graph reconstruction. Based on these re-compositional context-congruent scene graphs, RAS synthesizes images with Stable Diffusion models for replays. Unlike other image replay methods, RAS stores only triplet labels and the model snapshot, maintaining constant memory usage and preserving privacy. Extensive experiments demonstrate our RAS's superior performance over current CSEGG baselines in knowledge transfers and reducing forgetting. Interestingly, our RAS model is also capable of generalizing to classify known relationships among unseen objects.

Moving forward, there are several key avenues for future research. First, our current endeavors focus on tackling CSEGG problems from static images in an Independent and Identically Distributed (i.i.d) manner, diverging from how humans learn from video streams. Future research can look into CSEGG problems on video SGG datasets. Second, our plans also involve expanding the set of continual learning baselines and integrating more long-tailed distribution sampling techniques. Third, we aim to construct a synthetic SGG dataset to systematically quantify the aspects of SGG that influence continual learning performance under controlled conditions. In RAS, SGG annotations for synthesized images in the replay buffer are predicted by the preceding SGG model, which can lead to error propagation across training iterations. In the future work, integrating a generative model with fine-grained control signals (such as bounding boxes and captions) [36] could provide more precise supervision, potentially mitigating these accumulated errors and further enhancing the performance of our approach. Although the CSEGG method holds promise for many downstream applications like monitoring systems, medical imaging, and autonomous navigation, we should also be aware of its misuse in privacy, data biases, fairness, security concerns, and misinterpretation (see **Sec. A.8** for an expanded discussion). We invite the research community to join us in expanding and updating the safe use of CSEGG benchmarks, thereby fostering its advancements in research and technology.

Acknowledgement

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2021-025), and its NRFF award NRF-NRFF15-2023-0001. We also acknowledge Mengmi Zhang's Startup Grant from Agency for Science, Technology, and Research (A*STAR), Startup Grant from Nanyang Technological University, and Early Career Investigatorship from Center for Frontier AI Research (CFAR), A*STAR.

References

- [1] Somak Aditya, Yezhou Yang, Chitta Baral, Cornelia Fermuller, and Yiannis Aloimonos. From images to sentences through scene description graphs using commonsense reasoning and knowledge, 2015.
- [2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018.
- [3] Martin Andrews, Yew Ken Chia, and Sam Witteveen. Scene graph parsing by attention graph. *arXiv preprint arXiv:1909.06273*, 2019.
- [4] Frederik Benzing. Unifying importance based regularisation methods for continual learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2372–2396. PMLR, 2022.
- [5] Philipp Bomatter, Mengmi Zhang, Dimitar Karel, Spandan Madan, Claire Tseng, and Gabriel Kreiman. When pigs fly: Contextual reasoning in synthetic and natural scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 255–264, 2021.
- [6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020.
- [7] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248, 2018.
- [8] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 9516–9525, 2021.
- [9] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision (ECCV)*, pages 532–547, 2018.
- [10] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.
- [11] Zuyao Chen, Jinlin Wu, Zhen Lei, Zhaoxiang Zhang, and Changwen Chen. Expanding scene graph boundaries: Fully open-vocabulary scene graph generation via visual-concept alignment and retention. *arXiv preprint arXiv:2311.10988*, 2023.
- [12] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *Proceedings of the IEEE conference on computer vision and Pattern recognition*, pages 3076–3086, 2017.
- [13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

- [14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [15] Rui Gao and Weiwei Liu. Ddgr: continual learning with deep diffusion-based generative replay. In *International Conference on Machine Learning*, pages 10744–10763. PMLR, 2023.
- [16] Chandan Gautam, Sethupathy Parameswaran, Ashish Mishra, and Suresh Sundaram. Generative replay-based continual zero-shot learning. *arXiv preprint arXiv:2101.08894*, 2021.
- [17] Shalini Ghosh, Giedrius Burachas, Arijit Ray, and Avi Ziskind. Generating natural language explanations for visual question answering using scene graphs and visual attention, 2019.
- [18] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- [19] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019.
- [20] Muhammad Umair Hassan, Saleh Alaliyat, and Ibrahim A. Hameed. Image generation models from scene graphs and layouts: A comparative analysis. *Journal of King Saud University - Computer and Information Sciences*, 35(5):101543, 2023.
- [21] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- [22] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as compositions of spatio-temporal scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10236–10247, 2020.
- [23] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3668–3678, 2015.
- [24] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.
- [25] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017.
- [26] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.
- [27] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Mallocci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.
- [28] Stan Weixian Lei, Difei Gao, Jay Zhangjie Wu, Yuxuan Wang, Wei Liu, Mengmi Zhang, and Mike Zheng Shou. Symbolic replay: Scene graph as prompt for continual learning on vqa task, 2022.
- [29] Bo Li, Yongqiang Yao, Jingru Tan, Gang Zhang, Fengwei Yu, Jianwei Lu, and Ye Luo. Equalized focal loss for dense long-tailed object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6990–6999, 2022.
- [30] Lin Li, Jun Xiao, Guikun Chen, Jian Shao, Yueting Zhuang, and Long Chen. Zero-shot visual relation detection via composite visual cues from large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

- [31] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022.
- [32] Rongjie Li, Songyang Zhang, and Xuming He. Sgtr: End-to-end scene graph generation with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19486–19496, 2022.
- [33] Rongjie Li, Songyang Zhang, Bo Wan, and Xuming He. 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*, pages 11109–11119, 2021.
- [34] Yikang Li, Wanli Ouyang, Xiaogang Wang, and Xiao’ou Tang. Vip-cnn: Visual phrase guided convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1347–1356, 2017.
- [35] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE international conference on computer vision*, pages 1261–1270, 2017.
- [36] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22511–22521, 2023.
- [37] Yuanzhi Liang, Yalong Bai, Wei Zhang, Xueming Qian, Li Zhu, and Tao Mei. Vrr-vg: Refocusing visually-relevant relationships. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10403–10412, 2019.
- [38] Wentong Liao, Bodo Rosenhahn, Ling Shuai, and Michael Ying Yang. Natural language guided visual relationship detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.
- [39] Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. Beyond not-forgetting: Continual learning with backward knowledge transfer. *Advances in Neural Information Processing Systems*, 35:16165–16177, 2022.
- [40] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.
- [41] David Lopez-Paz and Marc’Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.
- [42] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 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*, pages 852–869. Springer, 2016.
- [43] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey, 2021.
- [44] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.
- [45] Angelo G Menezes, Gustavo de Moura, Cézanne Alves, and André CPLF de Carvalho. Continual object detection: A review of definitions, strategies, and challenges. *Neural Networks*, 2023.
- [46] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.

- [47] Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11321–11329, 2019.
- [48] Bryan A Plummer, Arun Mallya, Christopher M Cervantes, Julia Hockenmaier, and Svetlana Lazebnik. Phrase localization and visual relationship detection with comprehensive image-language cues. In *Proceedings of the IEEE international conference on computer vision*, pages 1928–1937, 2017.
- [49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.
- [50] Nishant Rai, Haofeng Chen, Jingwei Ji, Rishi Desai, Kazuki Kozuka, Shun Ishizaka, Ehsan Adeli, and Juan Carlos Niebles. Home action genome: Cooperative compositional action understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11184–11193, 2021.
- [51] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- [52] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.
- [53] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [54] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [55] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesaro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018.
- [56] Ivan Rodin, Antonino Furnari, Kyle Min, Subarna Tripathi, and Giovanni Maria Farinella. Action scene graphs for long-form understanding of egocentric videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18622–18632, 2024.
- [57] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.
- [58] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
- [59] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, page 1745–1752, USA, 2011. IEEE Computer Society.
- [60] Jeng-Lun Shieh, Qazi Mazhar ul Haq, Muhamad Amirul Haq, Said Karam, Peter Chondro, De-Qin Gao, and Shanq-Jang Ruan. Continual learning strategy in one-stage object detection framework based on experience replay for autonomous driving vehicle. *Sensors*, 20(23):6777, 2020.
- [61] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

- [62] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [63] Parantak Singh, You Li, Ankur Sikarwar, Weixian Lei, Daniel Gao, Morgan Bruce Talbot, Ying Sun, Mike Zheng Shou, Gabriel Kreiman, and Mengmi Zhang. Learning to learn: How to continuously teach humans and machines. *arXiv preprint arXiv:2211.15470*, 2022.
- [64] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. *Advances in neural information processing systems*, 26, 2013.
- [65] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.
- [66] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 398–414. Springer, 2022.
- [67] Jianren Wang, Xin Wang, Yue Shang-Guan, and Abhinav Gupta. Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10829–10838, October 2021.
- [68] Jianren Wang, Xin Wang, Yue Shang-Guan, and Abhinav Gupta. Wanderlust: Online continual object detection in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10829–10838, 2021.
- [69] Wenbin Wang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Exploring context and visual pattern of relationship for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8188–8197, 2019.
- [70] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022.
- [71] Chenshen Wu, Luis Herranz, Xialei Liu, Joost Van De Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.
- [72] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419, 2017.
- [73] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 670–685, 2018.
- [74] Kaiyu Yang, Olga Russakovsky, and Jia Deng. Spatialsense: An adversarially crowdsourced benchmark for spatial relation recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2051–2060, 2019.
- [75] Fei Ye and Adrian G Bors. Learning latent representations across multiple data domains using lifelong vaegan. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 777–795. Springer, 2020.
- [76] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.
- [77] Ruichi Yu, Ang Li, Vlad I Morariu, and Larry S Davis. Visual relationship detection with internal and external linguistic knowledge distillation. In *Proceedings of the IEEE international conference on computer vision*, pages 1974–1982, 2017.

- [78] Alireza Zareian, Zhecan Wang, Haoxuan You, and Shih-Fu Chang. Learning visual common-sense for robust scene graph generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 642–657. Springer, 2020.
- [79] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.
- [80] Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. Visual translation embedding network for visual relation detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5532–5540, 2017.
- [81] Ji Zhang, Yannis Kalantidis, Marcus Rohrbach, Manohar Paluri, Ahmed Elgammal, and Mohamed Elhoseiny. Large-scale visual relationship understanding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 9185–9194, 2019.
- [82] Ji Zhang, Kevin J Shih, Ahmed Elgammal, Andrew Tao, and Bryan Catanzaro. Graphical contrastive losses for scene graph parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11535–11543, 2019.
- [83] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. Regionclip: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16793–16803, 2022.

A Appendix

A.1 Introduction to Three Learning Scenarios

Within this section, we present more details of three learning scenarios and their practical applications.

A.1.1 Scenario 1 (S1): Relationship Incremental Learning

While existing continual object detection literature focuses on incrementally learning object attributes [43, 70, 8, 68, 60, 45], incremental relationship classifications are equally important as it provides a deeper and more holistic understanding of the interactions and connections between objects within a scene. To uncover contextual information and go beyond studies of object attributes, we introduce this scenario where new relationship predicates p_k are incrementally added in each task (**Fig. 2 S1**). There are 5 tasks in S1. To simulate the naturalistic settings where the frequency of relationship distribution is often long-tailed, we randomly and uniformly sample relationship classes from head, body and tail categories in Visual Genome [25], and form a set of 10 relationship classes for each task. Thus, the relationships within a task are long-tailed; and the number of relationships from the head categories of each task is of the same scale. To tackle this issue, we allow CSEGG models to see the same images over tasks, but the relationship labels are only provided in their given task

The design of such images and label splits over tasks aligns with human learning scenarios where a parent teaches the baby to recognize different toys and objects in the bedroom. Though the baby is exposed to the same bedroom scenes multiple times, the parent only teaches the baby to detect and recognize one object at a time in a continual learning setting. In the future, we will expand our studies to cases where the SGG models learn from non-overlapping sets of training images for each task. The same reasoning applies in **S2** and **S3**. Example relationship classes from each task and their distributions are provided in **Fig. S1**.

Here, we provide a concrete example application of Scenario 1 in medical imaging. Within medical imaging, an agent must acquire the ability to detect cancerous cells within primary tumors, like colon adenocarcinoma. Subsequently, it must extend this proficiency to identifying the same cell types within metastatic growths that manifest in different bodily regions, such as lymph nodes or the liver. In this instance, the identical cancer cell disseminates to fresh organs or tissues, progressively establishing new relationships with other cells over the course of time.

A.1.2 Scenario 2 (S2): Scene Incremental Learning

To simulate the real-world scenario when there are demands for detecting new objects and new relationships over time in old and new scenes, we introduce this learning scenario where new objects O_i and new relationship predicates p_k are incrementally introduced over tasks (**Fig. 2 S2**). To select the object and relationship classes from the original Visual Genome [25] for S2, we have two design motivations in mind. First, in real-world applications, such as robotic navigation, robots might have already learned common relationships and objects in one environment. Incremental learning only happens on less frequent relationships and objects. (2) Transformer-based AI models typically require large amounts of training data to yield good performances. Training only on a small amount of data from tail classes often leads to close-to-chance performances. Thus, we take the common objects and relationships from the head classes in Visual Genome as one task, while the remaining less frequent objects and relationships from tail classes as the other task. This results in 2 tasks in total with the first task containing 100 object classes and 40 relationship classes. In the subsequent task, the CSEGG models are trained to continuously learn to detect 25 more object classes and 5 more relationship classes. Same as **S1**, both the object class and relationship class distributions are still long-tailed within a task (**Fig. S1**).

Next, we provide two real-world example applications in robot collaborations on construction sites and video surveillance systems.

The CSEGG model's capacity to incorporate new objects and new relationships while retaining existing knowledge finds pivotal application in video surveillance contexts. Consider a company developing video-based security systems for indoor environments, capturing prevalent indoor objects and relationships. Expanding to outdoor settings like parking lots or restricted compounds demands retraining the model with new outdoor data alongside previous indoor data, ensuring operational effectiveness in both realms. The outdoor context introduces new objects like "cars" and relationships

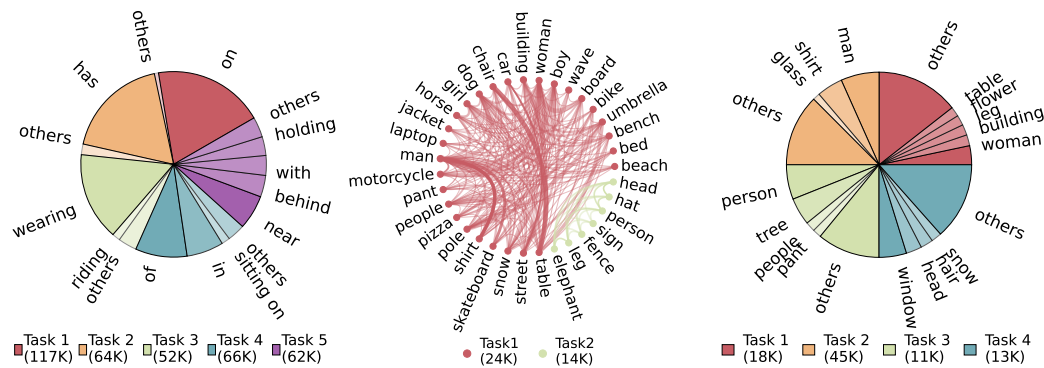


Figure S1: **Label distribution in each task in each learning scenario** is presented. In scenario S1 (a) and scenario S3 (c), we use different colors to denote different tasks. The color gradient indicates the frequency of data within a task, with the lighter color denoting the smaller frequency of data in that category. Only the most frequent labels (relationship labels in (a) and object labels in (c)) are provided. See the legend for the total data size per task. In (b) scenario S2 on both objects and relationships, data distributions are presented in the form of small-world networks, where nodes denote object categories and the edges linking object pairs indicate relationships. Thickness in edges implies the diversity of relationships between object pairs. Same color conventions as (a) and (c) are applied. See the legend for triplet sizes.

like "driving", distinct from indoor scenarios featuring "chair" and "sitting." Employing CSEGG allows the company to focus on new objects and relationships while retaining indoor insights.

Another real-world example would be a construction site where a team of robots is tasked with assembling various components to build a complex structure. Initially, during the foundation-laying phase, the robots are introduced to objects like "concrete blocks" and relationships like "stacking". As the construction advances to the wiring and installation phase, they encounter new objects like "wires" and relationships like "connecting," which were absent from earlier stages. The SGG model deployed in these robots needs to adapt incrementally to learn these new relationships without forgetting the existing ones. This ensures that the robots can effectively communicate and collaborate while comprehending the evolving scene and tasks, optimizing their construction efficiency and accuracy.

A.1.3 Scenario 3 (S3): Scene Graph Generalization In Object Incremental Learning

We, as humans, have no problem at all recognizing the relationships of unknown objects with other nearby objects, even though we do not know the class labels of the unknown objects. This scenario is designed to investigate whether the CSEGG models can generalize as well as humans. Specifically, there are 4 tasks in total with each task containing 30 object classes and 35 relationship classes. In each subsequent task, the CSEGG models are trained to continuously learn to detect 30 more object classes and learn to classify the same set of 35 relationships among these objects. The class selection criteria for each task follow the same as S1, where the selections occur uniformly over head, body, and tail classes. Example object classes and their label distributions for each task are provided in Fig. S1. Different from S1 and S2, a standalone generalization test set is curated, where the objects are unknown and their classes do not overlap with any object classes in the training set but the relationships among these unknown objects are common to the training set of every task. The CSEGG models trained after every task are tested on the same generalization test sets.

Here, we provide two real-world applications of Scenario 3 in the deep sea and space explorations for autonomous navigation systems.

A prime example is the ongoing research on deep sea exploration for autonomous navigation systems, where undiscovered flora and fauna reside beneath the ocean's surface. Encountering new and unidentified species becomes manageable through SGG's ability to understand spatial relations. The robot discerns the object's proximity or orientation even without precise identification of the species, enhancing its autonomous navigation ability. Likewise, in deep space exploration, SGG aids in recognizing spatial relationships with previously unseen space debris, aiding in path-planning. In

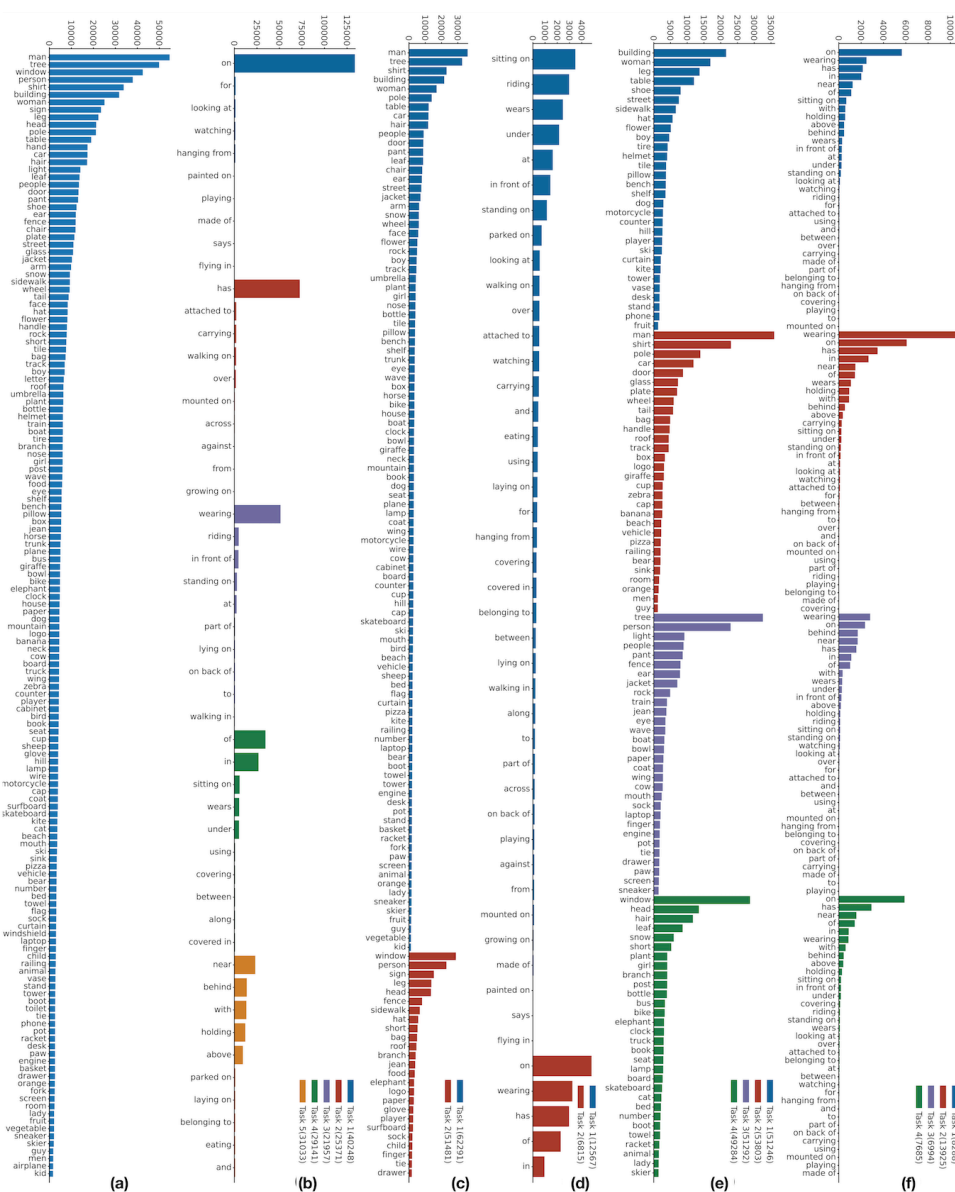


Figure S2: **Data Statistics for all Learning Scenarios:** (a) Distribution of objects in the entire training set of Visual Genome during Stage 1 of S1. (b) Distribution of relationships during Stage 2 for each task in S1. (c) Distribution of objects during Stage 1 for each task in S2. (d) Distribution of relationships during Stage 2 for each task in S2. (e) Distribution of objects during Stage 1 for each task in S3. (f) Distribution of relationships during Stage 2 for each task in S3. The numbers in brackets in the legend in (b-f) denote the number of training images in the particular task. Zoom in to the figure to get the exact labels and the frequency associated with them.

essence, SGG’s relationship generalization empowers robots to navigate and plan routes in unfamiliar terrains, such as deep sea and deep space, where novel encounters demand adaptable responses.

A.2 Data Statistics

In this section, we provide various types of data statistics for all three learning scenarios. Specifically, we present statistics regarding the number of images, objects, and relationships involved in each task of each learning scenario. This information is provided in Fig. S2.

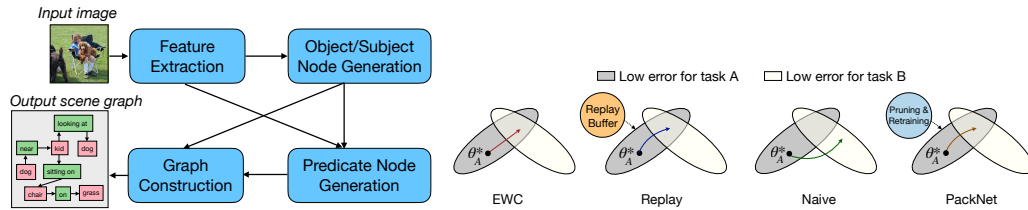


Figure S3: **Introduction to backbone SGG models and continual learning baselines.** We use Scene graph Generation TRansformer (SGTR) [32] as the SGG backbone (Sec. 3.2). SGTR consists of four modules indicated by each blue box. Arrows indicate the signal flows among modules. (b) Four continual learning baselines are listed: EWC [24], Replay [57], Naive (Sec. 3.2) and PackNet [44](Sec. 3.2). θ_A^* denotes the optimal network parameters after learning on task A. The arrows in colors indicate the shifts of network weights in the parameter space when learning Task B for different baselines.

A.3 Implementation and Training Details

A.3.1 SGTR Backbone

For SGTR in Fig. S3 (a), the approach uniquely formulates the task as a bipartite graph construction problem. Starting with a scene image (I_i), SGTR utilizes a 2D-CNN and transformer-based encoder to extract image features. These features are then incorporated into a transformer-based decoder, predicting object and subject nodes (O_i). Predicate nodes (R_i) are formed based on both image features and object node features, and a bipartite graph (G_i) is constructed to represent the scene collectively. The correspondence between object nodes (o_i) and predicate nodes (r_k) is established using the Hungarian matching algorithm [26]. Experimental results are based on the average over three runs, and the implementation leverages public source codes from [32] and [68] with default hyperparameters.

The SGTR is trained in two stages in a supervised manner. In stage 1, only object detection losses in DETR is [6] applied on O_i . In stage 2, only predicate entity prediction loss is applied on R_i , which can be further decomposed into L1 and GIOU losses for object/subject/predicate localization [54] and cross-entropy loss for object/subject/predicate classification. In learning scenario S1, we skip Stage 1, and directly load pre-trained weights of DETR for object detection on the entire training set of Visual Genome [32]. In stage 2 of S1, we freeze the feature extractor, and fine-tune the rest parts of SGTR for predicate entity predictions. As only relationship classes are incrementally introduced in S1, we freeze the entire weights of DETR for detecting all the objects in the scene over tasks. However, empirical results suggest that fine-tuning transformer-based encoders in DETR helps downstream predicate predictions [32]. Even with fine-tuning DETR in S1, we verify that there is minimal forgetting of detecting all objects in the scene over tasks (see Fig. S7 and Sec. A.6.2). Thus, the forgetting observed in S1 could only be attributed to incremental relationship learning. In Stage 1 of S2 and S3 where object classes are also incrementally introduced over tasks, we load weights of the feature extractor, pre-trained on ImageNet [13], and fine-tune the entire DETR [6] over all the tasks. Stage 2 of S2 and S3 is the same as S1.

Training the SGTR model involves two stages:

Object Detection Training: In this stage, a batch size of 32 is used. All methods are optimized using the Adam optimizer with a base learning rate of 1×10^{-4} and a weight decay of 1×10^{-4} . Object detection training is conducted only in the S2 and S3 scenarios. Each task in S2 is trained for 100 epochs, while each task in S3 is trained for 50 epochs. To expedite convergence, pre-trained weights on ImageNet are utilized before training on Task 1 for both S2 and S3.

SGG (Scene Graph Generation) Training: In this stage, the entire SGTR model is fine-tuned while keeping the 2D-CNN feature extractor frozen. A batch size of 24 is employed, and the Adam optimizer is used with a base learning rate of 8×10^{-5} . In S1 and S3, each model is trained for 50 epochs per task, while in S2, 80 epochs per task are used. All models are trained on 4 A5000 GPUs.

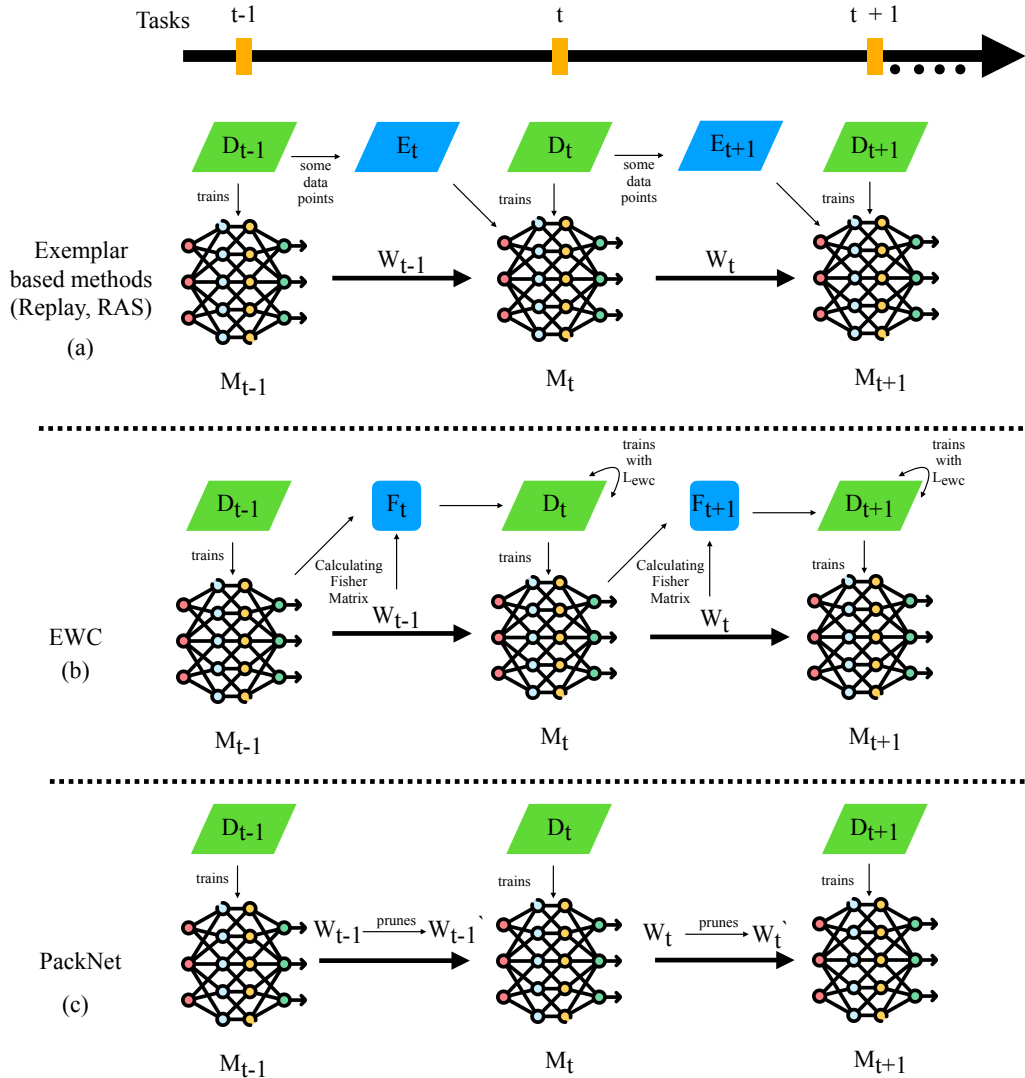


Figure S4: **Detailed Schematic of three CSEGG Baselines.** In this schematic, we show the operations of each baseline at tasks $t - 1$, t , $t + 1$. (a) shows the schematic for exemplar-based methods like replay and RAS. (b) shows the schematic for EWC. (c) shows the schematic for PackNet. Here, D_i refers to the dataset of task i . E_i denotes the exemplar set for replays at task i . W_i is the weights of M_i at task i .

A.3.2 TCNN Backbone

As for TCNN, it employs Faster-RCNN [18] to generate object proposals from a scene image (I_k). The model extracts visual features for nodes and edges from these proposals. Through message passing, both edge and node GRUs output a structured scene graph. Experimental results are based on the average over three runs.

Given a scene image I_i , TCNN utilizes Faster-RCNN[18] to generate a set of object proposals. The model subsequently extracts visual features of nodes and edges from the set of object proposals. Finally, both edge and node GRUs output a structured scene graph via message passing. The TCNN is trained in two stages in a supervised manner. In stage 1, only object detection losses in Faster-RCNN[53] are applied on O_i . We use the cross entropy loss for the object class and $L1$ loss for the bounding box offsets. In stage 2, the visual feature extractor (VGG-16[62] pre-trained on ImageNet [13]) and GRUs layers are trained to predict the final object classes, bounding boxes, and



Figure S5: **Visualizations of example generated images given the text prompt from our RAS.** Each row presents two examples of generated images. Given the input text prompt on the left, the generated image is displayed on the right. These visualizations provide qualitative validations into the capability of RAS to produce diverse and contextually relevant images based on our designed text prompts.

relationship predicates using cross-entropy loss and $L1$ loss. In Learning Scenario 1 (S1), similar to the implementation details of SGTR in **Sec. A.3.1**, we skip Stage 1, and directly load pre-trained weights of Faster-RCNN for object detection on the entire training set of Visual Genome [32]. In stage 2 of S1, we load the pre-trained weights of the visual feature extractor (pre-trained on ImageNet) and fine-tune the rest parts of the model. In stage 2 of S1, we load the pre-trained weights of visual feature extractor (pre-trained on ImageNet) and fine-tune the rest parts of the model. In Stage 1 of S2 and S3 where object classes are also incrementally introduced over tasks, we load weights of the Faster-RCNN, pre-trained on ImageNet [13], and fine-tune it over all the tasks. Stage 2 of S2 and S3 follows the same training regimes as Stage 2 of S1.

Object Detection Training: All methods are optimized using the SGD optimizer with a base learning rate of 1×10^{-2} and a weight decay of 1×10^{-4} . For training on the entire VG dataset, we train the model for 60 epochs with a batch size of 8 for both S2 and S3. To expedite convergence, pre-trained weights on ImageNet are utilized before training on Task 1 for both S2 and S3.

SGG (Scene Graph Generation) Training: A batch size of 12 is employed, and the SGD optimizer is used with a base learning rate of 1×10^{-2} and a weight decay of 1×10^{-4} . In S1, each model is trained for 30 epochs. In S2, each model is trained for 15 epochs. In S3, each model is trained for 25 epochs. All models are trained on 4 A5000 GPUs.

A.4 Mathematical formulations of Continual learning baselines

Here, we introduce mathematical formulations of these baselines: (1) To train Naive baseline on CSEGG, at task t , we take the previously trained model M_{t-1} with weights W_{t-1} and train it on the dataset D_t at task t , to obtain weights W_t of the model M_t . (2) For EWC, we use W_t and M_t to calculate F_{t+1} , where F is the Fisher information matrix using the equation, $F_{t+1} = -E[\frac{\partial^2}{\partial W_t^2} \log(M_t(x)|W_t)]$. During the training of task $t+1$, we add L_{EWC} to the training loss, where $L_{EWC} = \sum F_{t+1}(W_{t+1} - W_t)^2$. (3) For PackNet, after training of task t , we take W_t and apply a pruning algorithm to obtain the pruned weights W'_t . At task $t+1$, we obtain M_{t+1} by training M_t with W'_t on D_{t+1} . (4) For Replay@M, we create exemplar set E_t at the replay buffer by storing data points from D_{t-1} after training M_{t-1} on task $t-1$. At task t , we obtain weights W_t of M_t by training M_t on E_t and D_t . (5) For the joint training, we train one model M_T jointly on all the datasets $\{D_t\}$, where $t \in \{1, 2, \dots, T\}$. (6) For RAS_GT, we adopt the similar math formulations as Replay@M. See more details in **Sec. 4**.

A.5 Evaluation Metrics

To assess the catastrophic forgetting of CSEGG models, we define **Forgetfulness (F@K)** as the difference in $R@K$ on $D_{t=1}$ between the CSEGG models trained at task t and task 1. An ideal CSEGG model could maintain the same $R@K$ on $D_{t=1}$ over tasks; thus, $F = 0$ for all tasks. The more negative F is, the more severe in forgetting an model gets.

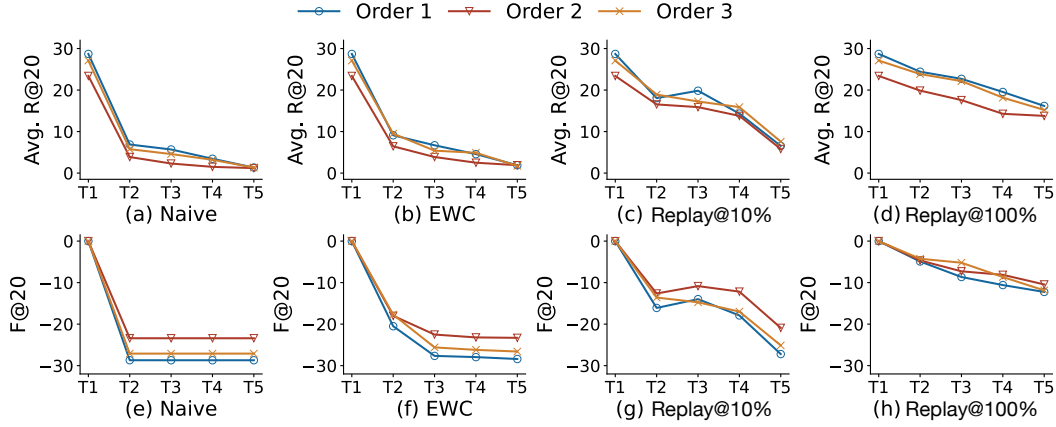


Figure S6: **Results of $F@K=20$, Avg. $R@K=20$ over tasks on CSEGG models with the SGTR backbone in Learning Scenario 1 with different permutations of task sequences.** (a),(e) denotes Avg. $R@K$ and $F@K$ for naive baseline. (b),(f) denotes Avg. $R@K$ and $F@K$ for EWC baseline. (c),(g) denotes Avg. $R@K$ and $F@K$ for Replay@10% baseline. (d),(h) denotes Avg. $R@K$ and $F@K$ for Replay@100% baseline. See Sec. 3.2 for introduction to continual learning baselines. See Sec. 3.3 for explanations about evaluation metrics. X-axis indicates the task numbers. The higher F , Avg. $R@20$, the better.

To assess the overall recall of CSEGG models over tasks, we also report the continual average recall (Avg. $R@K$). Avg. $R@K$ is computed as the average recall on all the data at the previous and current tasks D_i , where $i \in \{1, 2, \dots, t\}$.

To assess whether the knowledge at previous tasks facilitates learning the new task and whether the knowledge at new tasks enhances the performances at older tasks, we introduce **Forward Transfer (FWT@K)** [39] and **Backward Transfer (BWT@K)**[41]. BWT@K is defined as $BWT@K = \frac{1}{T-1} \sum_{i=1}^{T-1} R@K_{T,i} - R@K_{i,i}$, where T denotes the total number of tasks in a learning scenario and $R@K_{i,j}$ denotes the continual learning model trained after task i and tested in task j . FWT is defined as $FWT@K = \frac{1}{T-1} \sum_{i=2}^T R@K_{i,i} - \bar{b@K}_{i,i}$, where $\bar{b@K}_{i,i}$ is the $R@K$ for an independent model with random initialization trained in task i and tested in task i .

In learning scenario S3, we evaluate CSEGG models on their abilities to generalize to detect unknown objects and classify known relationships on these objects, in the standalone generalization test set over all tasks. To benchmark these, we introduce two evaluation metrics: the recall of the predicted bounding boxes on unknown objects (**Gen $R_{bbox}@K$**) and the recall of the predicted graph G_i (**Gen $R@K$**). As the CSEGG models have never been taught to classify unknown objects, we discard the class labels of the bounding boxes and only evaluate the predicted box locations with **Gen $R_{bbox}@K$** . To evaluate whether the predicted box location is correct, we apply a hard threshold of Intersection over Union (**IoU**) between the predicted bounding box locations and the ground truth boxes. If the IoU exceeds the predetermined threshold, it is considered a true positive (TP). We used IoU thresholds of 0.3, 0.5, and 0.7 for our experiments. If the IoU exceeds the threshold multiple times for the same predicted bounding box, we consider it a single positive prediction. This is because the metric aims to define the model's performance in locating bounding boxes of objects that have not been learned during training. Counting multiple times for the same box would be misleading, as it would inflate the number of TP s and recall, while the actual number of unknown bounding boxes the model can generate might be low. The total possible positives ($TP + FN$) are determined by the total number of ground truth boxes in the image. In general, **Gen $R_{bbox}@K$** helps evaluate how well the CSEGG model locates unknown objects within an image. True positives (TP) represent successful identification of the location of an unknown object, while $TP + FN$ represents all possible unknown objects the CSEGG model could locate. Thus, object classification labels are not required to calculate **Gen $R_{bbox}@K$** . Instead, we need the total number of ground truth bounding boxes in an image and the number of predicted boxes that meet the IoU thresholds.

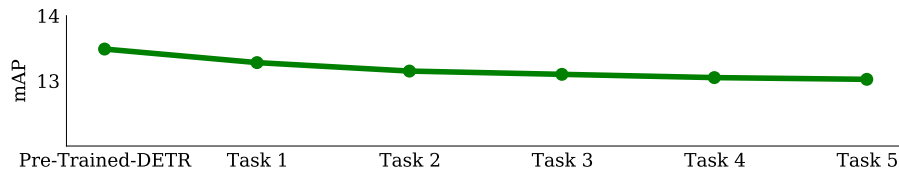


Figure S7: **mAP performance on the entire Visual Genome Test Set.** We used a pre-trained DETR checkpoint [32] as well as a naive baseline of SGTR continually trained on each task from S1, evaluated them on the entire Visual Genome's test set, and reported their mAP performances. We observed a minimal decrease in mAP across all tasks.

To assess whether the CSEGG model generalizes to detect known relationships over unknown objects, we evaluate the recall **Gen R@K** of the predicted relationships r_k only on *correctly predicted* bounding boxes.

A.6 More Result Analysis on Continual SGTR Methods

A.6.1 Results for Different Task Sequences

Recent work [63] has highlighted the consistent and substantial curriculum effects in class-incremental learning and continual visual question-answering tasks. Inspired by these findings, we conducted experiments to assess the impact of the curricula in the context of CSEGG. To delve into its potential influence, we trained baselines with three distinct task sequences for Learning Scenario 1. Our results demonstrated that curriculum learning indeed shapes CSEGG performance within class-incremental settings. Notably, in **Fig. S6(d)**, a large difference in Avg.R@20 between Order 1 and Order 3 emerges for the Replay (100%) baseline. Similarly, **Fig. S6(a)** reveals a substantial Avg.R@20 disparity between Order 2 and Order 3 for the Naive baseline. This trend extends to F@20, as depicted in **Fig. S6(e)(f)(g)(h)**. These insights collectively affirm the significance of the curricula within CSEGG.

A.6.2 Minimal Forgetting in DETR

To validate the impact of fine-tuning the DETR model in training Stage 2 of learning scenario S1 on relationship predicate predictions and to ensure minimal forgetting occurs in object detection (**Sec A.3**), we compare the mean Average Precision (mAP) for object detection on the entire test set of Visual Genome between the pre-trained DETR checkpoint from the paper [32] and the DETR models after fine-tuning on each task of S1.

As shown in **Fig. S7**, the results indicate a slight decrease of 0.4 in mAP from the pre-trained checkpoint to the DETR models over 5 tasks. This study provides evidence that fine-tuning DETR in S1 has negligible effects on forgetting. The forgetfulness observed in S1 can only be attributed to relationship incremental learning.

A.7 Visualization Examples for All Learning Scenarios for Continual SGTR-based Models

In this section, we present visualization examples from each learning scenario to showcase the performance of the three continual SGTR-based models, namely Replay@10%, EWC, and Naive, in three learning scenarios.

A.7.1 Learning Scenario 1 (S1)

From **Fig.S8** we observe that, in Task 1, the ground truth scene graph contains triplets of "on" relationship: "plate on table" and "hair on women". After training on task 1, all three models (Replay@10%, EWC, Naive) can accurately predict these triplets of "on" relationship.

In Task 2, triplets of "has" relationship are introduced: "plate has food" and "women has hair". After training on task 2 data, the Replay@10% model successfully remembers the triplets of "on" relationship ("plate on table", "hair on women") from Task 1 and predicts "women has hair". The Naive model forgets the triplets of "on" relationship and only predicts "women has hair". The EWC

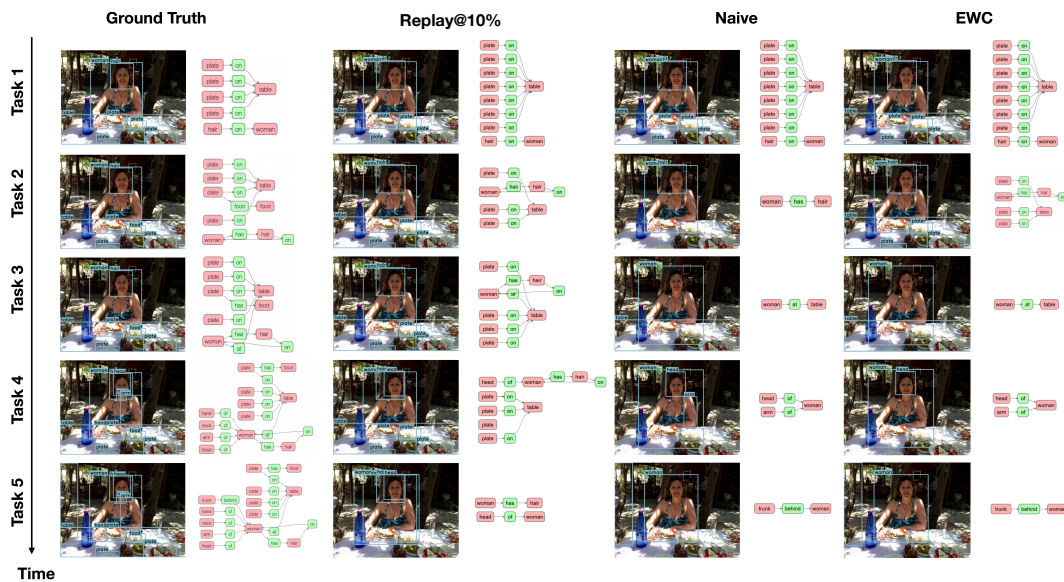


Figure S8: **Visualization example for Learning Scenario 1 (S1).** The leftmost column in the figure displays the ground truth bounding boxes and scene graphs for each task in Learning Scenario 1 (S1). The remaining columns, from left to right, represent the bounding boxes and scene graphs generated by each baseline model (Replay@10%, Naive, and EWC). In all the scene graphs, red boxes indicate objects, while green boxes represent relationships. The direction of the arrows between the red (object) and green (relationship) boxes indicates the subject and object ordering in the triplet. For example, in the scene graph predicted by the EWC model after Task 5, the triplet is "trunk behind women", as the arrow goes from "trunk" to "behind" to "women". The time arrow on the left side of the figure demonstrates that the model is exposed to new data over time, with new relationships incrementally added, as described in **Sec. 3.1**.

model remembers the triplets of "on" relationships and predicts "women has hair". None of the models predict "plate has food".

In Task 3, triplet of "at" relationship is introduced: "women at table". After training on task 3 data, the Replay@10% model remembers the previous triplets ("plate on table", "women has hair", "hair on women") and predicts "women at table". The Naive model forgets the previous triplets and only predicts "women at table". In contrast to its previous performance, the EWC model forgets the previous triplets and only predicts "women at table".

In Task 4, triplets related to "of" relationship are introduced: "hand of women", "neck of women", "arm of women", and "head of women". After training on task 4 data, the Replay@10% model remembers the triplets related to "on" and "has" relationships ("plate on table", "women has hair", "hair on women") from previous tasks but forgets the "at" relationship triplet ("women at table"). It only predicts "head of women" from the triplets introduced in Task 4. The Naive and EWC models both forget the "at" relationship triplet from the previous task but predict "head of women" and "arm of women" from the triplets introduced in Task 4.

In Task 5, triplet belonging to the "behind" relationship is introduced: "trunk behind women". After training on task 5 data, the Replay@10% model forgets the triplets related to "on" relationship ("plate on table", "hair on women") and only remembers the triplets related to "has" and "of" relationships ("women has hair", "head of women") learned from the previous task. It is not able to predict "trunk behind women". The Naive model, similar to its performance after previous tasks, fails to remember any triplets previously learned and only predicts "trunk behind women". The EWC model also fails to remember any triplets from the previous task and only predicts "trunk behind women".

A.7.2 Learning Scenario 2 (S2)

From **Fig.S9**, we observe that, in Task 1, the ground truth scene graph contains triplets: "man riding skateboard", "man above skateboard", and "shoe of skateboard". After training on task 1 data,

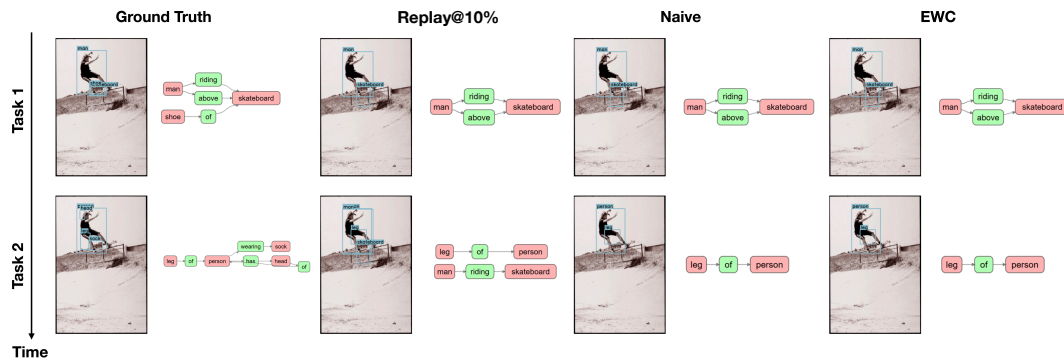


Figure S9: **Visualization example for Learning Scenario 2 (S2).** The leftmost column shows the ground truth bounding boxes and scene graphs in each task of Learning Scenario 2 (S2). The remaining columns, from left to right, represent the bounding boxes and scene graphs generated by each baseline model (Replay@10%, Naive, and EWC). In all the scene graphs, red boxes indicate objects, while green boxes represent relationships. As explained in Fig. S8 caption, the direction of the arrows between the red (object) and green (relationship) boxes indicates the subject and object ordering in the triplet. The time arrow on the left side of the figure demonstrates that the model is exposed to new data over time, with new objects and relationships incrementally added, as described in Sec. 3.1.

all three models (Replay@10%, Naive, EWC) predict “man riding skateboard” and “man above skateboard”. None of the models predict “shoe of skateboard”.

In Task 2, new triplets introduced are: “leg of person”, “person wearing sock”, “person has head”, and “head of person”. After training on task 2 data, the Replay@10% model only remembers “man riding skateboard” from the previous task, forgetting “man above skateboard”. Moreover, Replay@10% model can only predict “leg of person” from the triplets introduced in task 2. The Naive model forgets all the triplets from task 1 (“man riding skateboard”, “man above skateboard”) and only predicts “leg of person” from the triplets introduced in task 2. Similar to Naive model, EWC model forgets all the triplets from task 1 (“man riding skateboard”, “man above skateboard”) and only predicts “leg of person” from the triplets introduced in task 2.

A.7.3 Learning Scenario 3 (S3)

Fig. S10 illustrates the performance of the Replay@10% and Naive models in locating unknown objects and recognizing the relationships between these objects and other nearby unknown objects. The ground truth in **Fig. S10** consists of three unknown objects: “mountain”, “sheep”, and “house”, along with three relationships: “near”, “behind”, and “in front of” (between “mountain” and “house”), and a “near” relationship (between “sheep” and “house”).

After training on Task 1 data, the Naive model can accurately locate the objects “mountain” and “house”. However, no new object, such as “sheep”, is located by the Naive model after training on Task 2 data. After training on Task 3 data, the Naive model is also able to locate the object “sheep” in addition to “mountain” and “house”. Even after training on Task 4 data, the Naive model continues to locate all three objects: “mountain”, “house”, and “sheep”. In contrast, the Replay@10% model, after training on Task 1 data, can only locate “mountain” and “house”. This remains the same even after training on Task 2 and Task 3 data, where the Replay@10% model can still only locate the objects “mountain” and “house”. However, after training on Task 4 data, the Replay@10% model is able to locate all the objects: “mountain”, “house”, and “sheep”.

Regarding relationship generalization on unknown objects, the Naive model, after training on Task 1, can only predict the “near” relationship between the located objects “mountain” and “house” out of the three possible relationships. This performance remains the same even after Task 2. However, after training on Task 3, the Naive model can predict the “behind” relationship in addition to the “near” relationship between the located objects “mountain” and “house”. After Task 4, the Naive model can predict the “behind” and “near” relationships between the located objects “mountain” and “house”, as well as the “near” relationship between the located objects “sheep” and “house”.

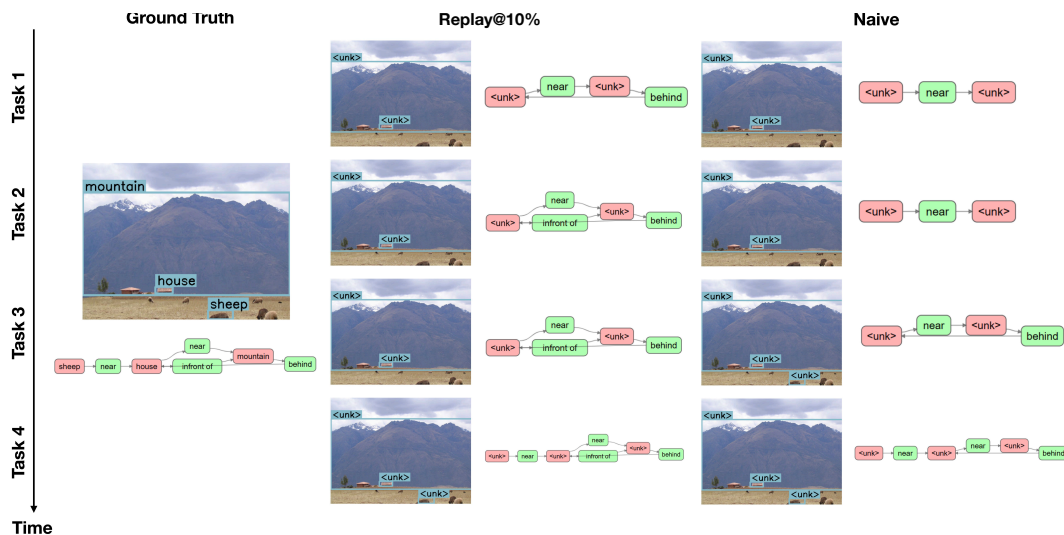


Figure S10: **Visualization example for Learning Scenario 3 (S3).** The leftmost column shows the standalone ground truth bounding boxes and scene graphs in the generalization test set regardless of which task it is in Learning Scenario 3 (S3). The remaining columns, from left to right, represent the bounding boxes and scene graphs generated by each baseline model (Naive, Replay@10%). Similar to **Fig. S8**, and **S9**, the red boxes in all scene graphs indicate objects, while green boxes represent relationships. As explained in **Fig. S8** caption, the direction of the arrows between the red (object) and green (relationship) boxes indicates the subject and object ordering in the triplet. The time arrow on the left side of the figure demonstrates that the model is exposed to new objects over time as described in **Sec. 3.1**. For easy referral of object instances in the predicted scene graphs, we numbered the unknown bounding boxes in this figure, where the numbers are not actually present in the model predictions. Concretely, unk_1 refers to "mountain"; unk_2 is "house"; and unk_3 is "sheep".

In contrast, the Replay@10% model, after training on Task 1, can predict the "near" and "behind" relationships between the located objects "mountain" and "house". After Task 2, it can also predict the "infront of" relationship between the located objects "mountain" and "house" along with "near" and "behind" relationships. Even after Task 3, the Replay@10% model is still able to predict the "near", "behind", and "infront of" relationships between the located objects "mountain" and "house". After Task 4, as it can now locate the object "sheep", the Replay@10% model can also predict the "near" relationship between the objects "house" and "sheep", in addition to the existing relationships between "mountain" and "house".

A.8 Ethical Concerns

The development and deployment of Scene Graph Generation (SGG) technology present potential negative societal impacts that warrant careful consideration [32]. Firstly, privacy concerns arise as SGG may inadvertently capture sensitive information from images, potentially violating privacy rights and raising surveillance issues. Secondly, bias and fairness challenges persist, as SGG algorithms can perpetuate biases present in training data, leading to discriminatory outcomes that reinforce societal inequalities. Misinterpretation and misclassification by SGG algorithms could result in misinformation and incorrect actions, impacting decision-making. The risk of manipulation and misuse of SGG-generated scene representations for malicious purposes is also a concern. For example, attackers might manipulate scene graphs to deceive systems or disrupt applications that rely on scene understanding.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: **Sec. 5**

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: **Sec. 6**

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification: We don't have any theoretical proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: **Sec. A.3**

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [\[Yes\]](#)

Justification: We have provided the code as the Supplementary material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: **Sec. A.3**

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[Yes\]](#)

Justification: In **Sec. A.3**, we have provided information about the statistical significance of the experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: **Sec. A.3**

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification: We have followed the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: **Sec. A.8**

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [\[Yes\]](#)

Justification: We have used Stable Diffusion models for the image generation. So we have ensured to put their safeguards during the image generation process.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [\[Yes\]](#)

Justification: We have credited the authors of the models and methods we are using for our work.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [\[Yes\]](#)

Justification: **Sec. 4**

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [\[NA\]](#)

Justification: We have not done any crowdsourcing or research involving human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [\[NA\]](#)

Justification: We have not done any crowdsourcing or research involving human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.