OG-SGG: Ontology-Guided Scene Graph Generation. A Case Study in Transfer Learning for Telepresence Robotics

Fernando Amodeo *1, Fernando Caballero †2, Natalia Díaz-Rodríguez ‡3, and Luis Merino §1

1Service Robotics Laboratory, Universidad Pablo de Olavide, Seville, Spain.
2Service Robotics Laboratory, Universidad de Sevilla, Spain.
3Computer Science and Artificial Intelligence Dept., Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, Spain

Abstract

Scene graph generation from images is a task of great interest to applications such as robotics, because graphs are the main way to represent knowledge about the world and regulate human-robot interactions in tasks such as Visual Question Answering (VQA). Unfortunately, its corresponding area of machine learning is still relatively in its infancy, and the solutions currently offered do not specialize well in concrete usage scenarios. Specifically, they do not take existing “expert” knowledge about the domain world into account; and that might indeed be necessary in order to provide the level of reliability demanded by the use case scenarios. In this paper, we propose an initial approximation to a framework called Ontology-Guided Scene Graph Generation (OG-SGG), that can improve the performance of an existing machine learning based scene graph generator using prior knowledge supplied in the form of an ontology; and we present results evaluated on a specific scenario founded in telepresence robotics.

1. Introduction

Telepresence robots allow people to remotely interact with others. They are often called “Skype on a stick” because they combine the conversation capabilities of teleconference software with the mobility of robots controlled by humans for a better social interaction [22]. They are also sometimes referred to as “your alter-ego on wheels” because they have a clear application in assistance tasks. For example, they allow disabled people to attend events remotely, or caregivers to interact remotely with people under their care. In particular, we consider in this work the application of telepresence robots for elderly care [56] (see Fig. 1). However, controlling these systems is a complex task. The human user needs to focus on both low-level tasks (such as controlling the robot) and high-level tasks (such as maintaining a conversation) at the same time; and this can lead to a cognitive overload, therefore reducing the attention that is given to the high-level tasks [41]. One approach to reduce this overload involves leveraging semi-autonomous capabilities to allow the user to control the robot using only high-level commands (i.e., Approach a given object, Follow a given person, etc), while the robot takes care of low-level control. This is in fact a necessity if we consider visually-impaired people as users of the robotic system.

In all these cases, the robot needs to extract and provide semantic information about the scenario so that the scene can be described in human terms to the users, and they can in turn indicate the robot where to go next for interactions. At the same time, we need a representation of this information that allows the robot to perform automated reasoning.

In this work, we consider recent advances in visual scene graph generation (i.e., [44, 37, 20, 48]) and investigate their application to telepresence robots. These methods extract a semantic graph for a given image, composed of the main objects present in the scene and relations between them. The main problem with current solutions is that they are aimed at general scenarios and do not take existing “expert” knowledge about the domain world into account; and that might indeed be necessary in order to provide the level of reliability demanded by the usage scenarios.

The main goal of our work is thus finding a way to reuse and repurpose existing scene graph generation models and datasets
for specific robotic applications, and applying additional techniques that take into account existing domain knowledge of the application, so that we can improve the performance of a machine learning model within the reduced scope of a given problem and ontology. Our aim is proposing a protocol that can describe scenes using ontology-founded scene graphs, and validating it in a real transfer learning case for ontology-based generation.

In order to justify our approach, we showcase a small specialized dataset and ontology founded in our robotics application that we later use as a benchmark.

2. Background

Ontologies are broad constructs that can be used to represent the cognitive model of a given domain world [13]. In simplified terms, we can explain that an ontology defines a class hierarchy of objects that can exist in the world, as well as the different types of relations between the objects of the world (called predicates). Most importantly, an ontology is able to define axioms that restrict how the predicates can be applied, in addition to producing implicit, reasoned knowledge from a set of assertions made within the scope of the ontology. For this reason they are the tool of choice to represent knowledge bases in robotics and other fields [17][50] and perform context-awareness reasoning [2][13]. Ontologies can also be used to model knowledge graphs with richer and more formal semantics, allowing for higher order reasoning.

We based our work on OWL 2 [42], the standard knowledge representation language for defining ontologies created by the W3C. OWL is built upon RDF [19], which is an earlier W3C XML standard with the purpose of facilitating data interchange on the Web. Ontologies are a suitable tool to achieve explainable ML models [17] in the form of knowledge graphs and other semantic web technologies [54].

Given a scene and an ontology, we can build a scene graph by defining a set of objects \( O = \{ o_1, o_2, \ldots, o_n \} \) (from the classes defined in the ontology) that appear within it, along with a set \( R \) of asserted relation triplets \( \{ (o_i, p, o_j) \} \), where \( o_i \) is the source object of the relation, \( o_j \) is the destination object, \( i \neq j \), and \( p \) is the predicate that describes the relation. In addition, we can define \( P_{ij} = \{ pk \mid (o_i, pk, o_j) \in R \} \), which is the set of predicates for which a corresponding relation triplet exists along the object pair \( (o_i, o_j) \). Since there are \( n \) objects in the scene, we can conclude that there are \( n(n-1) \) object pairs, each with an associated \( P_{ij} \) predicate set.

A scene graph generator is a system that, given an input corresponding to a particular scene (most often an image together with object detection information), predicts the contents of the \( P_{ij} \) predicate set for all given \( (o_i, o_j) \) object pairs. There are several ways to implement a scene graph generator, including classical methods based on hardcoded rules, but the most promising area of research nowadays involves neural networks based on supervised deep learning. We considered this approach in our paper.

Scene graph datasets are collections of annotated scenes (images) intended for evaluating scene graph generators, as well as training the aforementioned scene graph generation networks. Each image in the dataset is annotated with its associated set of objects \( O \) and set of relation triplets \( R \). In particular, \( O \) is usually annotated as a series of bounding boxes with class information, each corresponding to an object; while \( R \) is annotated as a list of \( (o_i, o_j) \) object pairs with their corresponding \( P_{ij} \) predicate set.

A common approach to building scene graph generation networks involves predicting a ranking score for all possible relation triplets across the entire image, where higher ranking triplets are deemed as more likely to occur than lower ranking triplets. A trimming operation takes place afterwards, which removes triplets according to a given set of criteria. The conventional way to define this operation is Top-K, which results in the \( K \) highest ranking triplets being retained while the rest are discarded.

In this work we propose a series of techniques to improve this pipeline by involving axioms defined in the ontology that govern the semantics of the predicates. These axioms are used to augment the source data used during training and control the trimming operation that affects the network’s output. We considered the following types of axioms that affect predicates, which are defined by [42]:

- **Domain and range restrictions**: these axioms assert that only objects belonging to certain classes can be the source or destination of a predicate (respectively). For example, we can say that for the predicate sitting on, the domain is Person and the range Chair – it is not possible to say (plant, sitting on, food).
- **Inverse relationships**: these axioms assert that one predicate is the inverse of another (with the source and destination objects inverted). For example, we can say that the predicates on top of and below are inverses of one another.
- **Transitivity**: these axioms assert that, if two relation triplets \( (o_i, p, o_j) \) and \( (o_j, p, o_k) \) are given, then \( (o_i, p, o_k) \) also holds. For example, we can say that the predicate behind is transitive, since if both (person, behind, chair) and (chair, behind, table) hold, then (person, behind, table) must also hold.
- **Functionality**: these axioms assert that there can only be one object related by a predicate to a different one. For example, the predicate holding can only accept one source object for each destination object – if (person, holding, pencil) holds, then no other Person object can be related to pencil through holding.
- **Symmetry**: these axioms assert that a certain predicate does not mandate an order in which the two objects are related. This means that if the source object is related to the destination object through the predicate, then the destination object is also related to the source object through the same predicate. For example, (chair, next to, table) implies (table, next to, chair).
3. Related work

Given our use case in telepresence robotics, we initially investigated simpler, more direct approaches such as automatic image captioning [46], combined with refinement based on data sampled from our own robot. The main problem with this approach had to do with the lack of structure and lack of coherence in generated captions (which we confirmed using indicators such as Semantic Fidelity [11]), resulting in unsatisfactory results from the point of view of potential users, as well as lack of usability for downstream robotic tasks.

We quickly learned that a more formal and richer way of representing knowledge about environments was necessary. This led us to investigate scene graphs, specifically existing work concerning their automatic generation. Another justification for our interest in scene graphs is their potential for usage in neuro-symbolic (NeSy) computation approaches. Such approaches are increasingly being used to improve the explainability of AI solutions (XAI) [14]. In this section we detail our survey of the state of the art, as well as what we believe to be their relevance and contribution to solving domain specific problems like ours.

3.1. Scene graph datasets

The main ingredient that makes or breaks a machine learning application is the dataset. A good dataset, although not a guarantee of success, is a necessary precondition. Currently there exist several of them which have some relevance to our domain-specific scene graph generation task:

- **MS COCO** [26] is the de-facto standard dataset for image classification, segmentation, captioning and object detection. However, despite offering 5 free-form textual captions per image, it does not contain any usable semantic information needed to make a machine learn how to generate scene graphs. Nonetheless, other researchers have built upon MS COCO in order to create scene graph datasets, such as those which will be discussed next.

- **Semantic PASCAL-Part** [12] is an OWL conversion of an earlier PASCAL-Part dataset [8]. Formal ontological constructs are used to define each object class, which makes this dataset appropriate for object classification tasks based on detection of constituent parts. Unfortunately, this also means that the dataset is not suitable either for downstream scene graph tasks, as there is effectively only one possible “relation”: isPartOf (and its inverse hasParts).

- **VRD** [24] (Visual Relationship Detection) is one of the first datasets developed during scene graph generation research. It is a relabelling of an earlier Scene Graph dataset [18], which was itself sampled from the intersection of MS COCO and YFCC100m [39].

- **VG** [21] (Visual Genome) is a follow-up work to VRD that opens up the annotations to cover the entire intersection between MS COCO and YFCC100m instead of a hand picked sample. It was created by crowd sourcing, and it brings additional ground truths such as relations between the objects or visual question answering examples. It makes use of WordNet [28] to identify objects and relations, which adds considerable depth to the labelling compared to MS COCO with its 80 broad categories. However, since the semantic data is generated automatically from the crowd sourced input, it is quite noisy and thus requires serious preprocessing before it can be used. The maintainers of VG also offer a list of duplicate/aliased object and relation classes that is nearly always used as the first step of the required preprocessing.

- **VG-SGG** is a preprocessed version of VG introduced by [45] which has subsequently been adopted by researchers as the VG split of choice for training and evaluating scene graph generation networks, hence the name. Bounding box information is cleaned up, and only the 150 most frequent object classes and 50 predicate classes are used.

- **VrR-VG** (Visual-relevance Relations) is another filtered and improved version of VG specifically intended for scene graph generation. It improves VG by removing from the dataset high frequency, low quality ambiguous relations that can be easily detected with mere probabilistic analysis; and leaving smaller, high quality ones that require visual and semantic reasoning to detect.

- **GQA** [16] (Graph Question Answering) is yet another dataset based on VG, but focused on visual question answering. Even though it is intended to be used to solve a different task it is still of interest, because it contains scene graphs with object information that has been further

---

*Figure 2: An image captured by the robot, annotated with a scene graph.*

[http://visualgenome.org](http://visualgenome.org)
cleaned, filtered and even manually validated. In addition, it augments images with a location annotation that discloses the type of environment (indoors or outdoors).

Overall, the scope of existing datasets tends to be very broad, aiming to fulfill the general use case (without any specific domain in mind). In addition, they exhibit great imbalance of object and relation classes, which makes it more difficult to train domain-specific models, as well as resulting in undesirable bias towards the few most overwhelmingly common classes. Furthermore, the labels are usually noisy and sparse (incomplete), which results in a risk of the network learning fictitious unintended patterns. Finally, none of the existing datasets have a formally defined ontology, instead relying on free form annotation of objects and relations devoid of any specific semantics that could be used to extract additional inferred knowledge, or improve the quality of the generated graphs.

In this work, we propose methods by which existing scene graph datasets can be adapted to fit within an existing ontology, as well as enriching the annotations through inferred knowledge derived from the axioms in the ontology. The desired end goal of this endeavor is making it possible to reuse existing data to solve new domain-specific scene graph generation problems.

### 3.2. Scene graph generation

Researchers have been iterating over different ideas on how to approach the scene graph generation problem. Below is a summary of some of the most interesting approaches that have been published:

- The original VRD model [27] introduced alongside the dataset proposed a simple network based on two modules that looked at visual and language features respectively, and incorporated likelihood priors based on the predicate frequency distribution for a given pair of object classes.
- Iterative Message Passing [25] proposed using RNNs to iteratively "pass" information between proposed edges of the graph in order to further refine them using their neighboring context.
- Neural Motifs [48] introduced a new architecture based on bidirectional LSTMs that was capable of detecting patterns in the structure of the scene graphs called "motifs".
- VRD-DSR [24] proposed combining visual appearance, spatial location and semantic embedding "cues" in a single network, as well as treating scene graph generation as a triplet ranking problem.
- Unbiased Causal TDE [37] proposed a new scene graph benchmark framework with better defined metrics, along with a new model agnostic technique that aims to reduce bias during training.
- Schemata [35] were proposed as a way of introducing an inductive bias in the form of relational encoding that allows the network to learn better representations from the training data as non-expert prior knowledge, resulting in better generalization. This encoding can also be propagated during model fine-tuning with additional external triplet data, without the need for image data.
- VRD-RANS [44] improved upon VRD-DSR by changing the visual feature extraction, adding a recursive attention module with a GRU, and integrating a form of data augmentation based on negative sampling into the training pipeline.
- RTN [20] applies the Transformer architecture to scene graph generation for the first time. It follows a conservative approach in regards to the inputs used and its usage of a likelihood prior, and introduces the concept of positional embedding for nodes and edges.

In general, analogously to the corresponding work on creating datasets, these models aim to solve the general problem of scene graph generation. Most if not all models are based around a traditional two-stage object detector such as Faster R-CNN [32] (close, but not quite usable in realtime), and many make use of the intermediate feature maps extracted for specific regions of interest prior to the object classification stage. In addition, existing codebases are geared towards training and evaluating the models (including the object detectors) on existing datasets, with little to no thought put into transfer learning tasks, custom datasets, or evaluations of individual components. These factors make it needlessly difficult to adapt existing scene graph generation solutions to new problem domains such as our robotics application.

On the other hand, the methods we propose to improve the usability of scene graph generation for specific domains are independent of the model used. That is, they can be adapted for use with any specific scene graph generation model that might be best suited within the constraints imposed by the problem (such as efficiency, for example).

### 3.3. Robotics related research

Several works with some relevance to our problem have previously been published, including the following:

- Ontologenius [33] is a semantic memory module based on OWL ontologies for the ROS [31] environment. It can be used to store the knowledge of the robotic agent, as well as to perform reasoning on it. However, it does not contain any perception functionality, meaning it needs to be supplied externally with knowledge by other nodes within the ROS environment. Some research papers such as [6] make use of it as the underlying knowledge engine.
- RoboSherlock [7] is a framework for cognitive perception based on unstructured information management. It offers perception related functionality, but the implementation is based on classical algorithms instead of deep neural networks, which makes it difficult to generalize to new environments, situations or use cases.
- Other research such as [29, 10] makes use of deep learning models to construct scene graphs in robotic contexts, but their approach is also limited in scope and application. In addition, the potential for using the internal structure of the ontology to guide the process is left untapped; instead still relying on labels devoid of any semantics, i.e. without a formally defined ontology that describes the class hierarchy and axioms governing the predicates.

Overall, there are interesting building blocks that can be used as reference for developing new applications. However, the us-
age of ontologies for semantic perception is still fairly young. Moreover, researchers prefer placing their focus on solving specific problems by sacrificing the potential for generality. In our work, even though we showcase a specific application, we aim to provide methods that can be reused in other applications without significantly changing how they work.

4. Methods

Our proposed pipeline, illustrated in Figure 3, consists of three main components: a scene graph generation network, a training dataset filtering and augmentation process, and a network output post-processing process. These last two processes, the core of our contribution, make use of pre-existing expert knowledge defined in the domain ontology, while the network itself can be adapted from the existing state of the art with minimal changes according to needs (such as efficiency).

4.1. Scene graph generation network

We settled on VRD-RANS [44] as the baseline scene graph generation network for our research (illustrated in Figure 4), which we implemented as faithfully as possible. This network was chosen for the following reasons:

- It uses information about the objects within the scene in the form of semantic vectors. Each object class is assigned a different semantic vector, in turn sourced from an existing corpus of pre-trained word embedding vectors. This is intended to improve the generalization capability of the network and allows it to be repurposed for different sets of possible input objects without needing retraining.
- It only needs a single global feature map extracted from the image, as opposed to other solutions which mandate the use of two-stage object detectors based on Regions of Interest (RoIs). This allows for using quick, robotics oriented single-stage object detectors such as those belonging to the YOLO or SSD families.
- It receives object localization information in the form of binary masks, which could be expanded in future work to contain additional information captured by the robot.
- It contains a recursive attention module, allowing the network to focus on processing the most relevant parts of the image at once.
- It uses a novel training strategy that consists of providing the network with fixed-size batches, one for each image in the dataset, and containing examples of both labelled and unlabelled object pairs (referred to as positive and negative examples, respectively). The idea behind this is compensating for the sparse nature of datasets, and taking advantage of the large number of unannotated pairs for data augmentation and regularization purposes.

VRD-RANS, like other scene graph generation networks [45] [48] [37], operates on an object pair by pair basis (each individual corresponds to a given object pair), and is in charge of predicting ranking scores for each of the predicates defined by the scene graph dataset. These ranking scores are raw unbounded values $\in \mathbb{R}$ with no defined semantics other than comparison operators (i.e. $\lt$, $\gt$, $\leq$, $\geq$, $=$, $\neq$). Therefore, in order to generate a scene graph, all object pairs must be fed to the network and the resulting triplet ranking scores are sorted in descending order. The threshold below which triplet proposals are judged as more unlikely than likely is left undefined, therefore requiring users of the network to establish their own post-processing rules. In a later section of this paper, we will revisit this area, where we propose post-processing rules that take into account information expressed in the ontology.

As in VRD-RANS [44], the loss function used to train the network is the the multi-label hinge loss margin function. The following scalar loss value is calculated, cross referencing all object pairs that appear within the training minibatch:

$$L = \frac{1}{NN} \sum_{i (y_i = 0)} \sum_{j (y_j = 1)} \max(0, 1 - (\hat{y}_j - \hat{y}_i))$$

where $N$ is the number of object pairs in the minibatch (i.e. its size), while $n$ is the number of predicates (i.e. network outputs). Thus, $NN$ is the total number of triplet predictions in the minibatch. $y_i$ is the ground truth value for a given triplet $i$ in the minibatch (evaluating as 1 if the triplet is present and 0 if not present), while $\hat{y}_i$ corresponds to the output of the network (ranking score). This function, which takes the entire minibatch output of the network at once, is thus designed to cause the network to incur a loss when the scores $\hat{y}_i$ corresponding to triplets not present in the ground truth ($i \mid y_i = 0$) are ranked higher than those which are present ($j \mid y_j = 1$).

4.2. Dataset filtering and data augmentation

We decided to focus our work on reusing existing scene graph datasets and repurposing them to be usable within the scope of our problem, which consists in describing a scene with a knowledge graph. For this, we defined a formal ontology for the target problem, and applied a series of ontology-guided transformations to the source dataset.

First of all, we parse the original source format of the input data and convert it to a common representation. We also convert source images to feature maps. During this step, an initial form of filtering based on ad-hoc constraints can be carried out (such as, for example, removing all images not tagged in a particular way, e.g. “indoor” images; or not containing specific objects, and so on). This has the (possibly desired) side effect of reducing the size of the dataset while maximizing or maintaining its quality in terms of performance. In a later section we will show how the filtering can sometimes even lead to improvements in the results.

Next, it is necessary to convert object class annotations into semantic vectors that can be fed to the network. We decided to use an existing word embedding model pre-trained on the English Wikipedia corpus [3] in order to generate the semantic vectors. Mapping objects from the source set of classes to the ontology’s set of classes was deemed unnecessary, given the

https://tfhub.dev/google/Wiki-words-250-with-normalization/2
The generalization power of training the network on a much richer set of semantic vectors than the one that can be derived from the reduced set of classes in the ontology.

Following that, predicates defined by the source scene graph dataset need to be mapped into the corresponding predicates of interest defined by our domain ontology. In order to do this, we manually defined the correspondence between the two sets of predicates, which is then used during this process to translate the predicate component of each relation triplet. Additionally, we discarded relation triplets that contain predicates not matched with any in the ontology. This mapping is the only information that needs to be externally defined and provided during the process, besides the ontology itself.

Once all triplets are using predicates defined in the ontology, we feed each scene in the dataset to an OWL processor module previously initialized with the ontology. We selected Owlready2 as the software library providing ontology processing. This ontology processor is able to load and parse ontologies in the OWL format, perform inferences on the provided knowledge using the axioms defined in the ontology, and generate implicit triplets. This includes the generation of triplets for inverse, symmetric, and transitive predicates. For example, \( (\text{chair}_1, \text{next to, table}_1) \) would generate \( (\text{table}_1, \text{next to, chair}_1) \). Likewise, \( (\text{cup}_1, \text{on top of, table}_1) \) would generate \( (\text{table}_1, \text{below, cup}_1) \). These new triplets are then extracted from the ontology processor and added back to the dataset, thus resulting in a form of data augmentation.

Finally, once the enriched information obtained through ontological inferences is extracted from the ontology processor, the final combined data can be divided into training and validation splits. Note that the test split of the original dataset is never used, as we are only interested in converting training data. We set up this arrangement in order to tune the hyperparameters of the model and implement early stopping in the training process. Stratification is applied in order to preserve the frequency distribution of predicates, which needs to be as similar (and complete) as possible between the two splits. In order to stratify a multi-label multi-class data structure such as scene graphs, we decided to first tally how many images a given predicate appears in, and then assign the images into buckets corresponding to the least frequent predicate classes that appear in each. Afterwards, the bucket for the overall least frequent predicate can be selected and its images added to the stratified splits according to the desired proportion. Since we removed images from circulation by doing this, the frequency distribution must be recalculated and the bucket assignment redone. We repeat this process until all predicates are processed.

### 4.3. Output postprocessing

The axioms defined in the ontology restrict the set of possible relation triplets that can appear in a scene, and thus can be used to filter out predictions that we know beforehand to be invalid.

We propose appending a new output post-processing stage to the prediction process that discards relation triplets that introduce violations of axioms defined in the ontology. We studied the leveraging of several kinds of axioms that affect predicates, such as Functional/InverseFunctional restrictions, or domain/range restrictions. The general filtering approach consists of only accepting the highest ranking mutually exclusive triplet proposals and discarding the rest. For example, if there are two triplet proposals, \( (\text{person}_1, \text{sitting on, chair}_1) \) with score 0.78 and \( (\text{person}_1, \text{sitting on, chair}_2) \) with score −0.4, the first triplet is accepted and the second one is discarded.

Additionally, we decided to study domain/range axioms, in particular using the internal semantic structure of the relationships between object classes in the ontology. We express the
domain/range axioms as a boolean tensor $C$ with the shape $|O| \times |O| \times |P|$, where the first two dimensions correspond to the object classes in the ontology (i.e. a generic pair of objects) and the last dimension to the predicate classes – this unusual ordering is used in order to improve the efficiency of the retrieval of predicate compatibility information according to a given object pair used as key. An element of the tensor is True if its associated predicate is compatible with the domain and range corresponding to the given object classes, and False otherwise. Predicted triplets from the output of the model can thus be individually looked up in $C$ and kept or discarded according to the truth value.

We compute $C$ beforehand by programmatically introspecting on the ontology, matching up all possible pairs of object classes, and verifying their compatibility with all predicate classes. We implemented support in the introspection code for a basic set of logic constructs found in OWL that are used to define the domain/range of a predicate. This includes the And, Or and Not operators; as well as support for walking through the class hierarchy (e.g. if the range of a property is GrabbableObject, then it will be legal to have a Cup in the object position).

4.4. Implementation details

We selected YOLOv4 [5] as the object detection network of choice given its high real-time performance, suitable for robotics applications. We used existing weights pretrained on MS COCO [26], which include a CSPDarknet53 [43] backbone pretrained on ImageNet [11].

Given that no code was provided, we decided to reimplement [44]. Our implementation was carried out using the TensorFlow framework, with the high-level Keras API layered on top. We created a custom model subclass with an overridden `train_step` method that implements the fixed batch size object pair sampling process explained in [44]. We chose not to implement sampling probabilities or statistical priors unlike [44], in order to maximize the zero-shot performance of the model on fully unseen data – these priors can also be seen as a way of artificially boosting performance by knowing beforehand that the test split is sourced from the same dataset as the training split, and thus both splits share similar statistical properties. In our case this is counter-productive, as we are precisely trying to apply the system to a transfer learning problem. The hyperparameters we used are listed in Table 1. The network was trained on a single NVIDIA Quadro RTX 5000 GPU.

5. Evaluation protocols for scene graph tasks

Models dealing with scene graphs are known to be difficult to evaluate. There exist several different tasks to evaluate them.
on, and it is necessary to deal with problems arising from the incomplete/noisy/biased nature of the datasets. In this section we detail the methodology we devised to evaluate different approaches, as well as the challenges we faced.

First of all, existing work [37,48] considers and evaluates the following three tasks separately, under various different names:

- **Predicate Detection (PredDet):** Given an image and a list of objects in it (with bounding boxes and class information), rank all candidate relation triplets that can form a Scene Graph between the objects. This is the simplest version of the task and it is intended to only specifically evaluate the reliability of the relation detection. We chose to focus on evaluating this task.

- **Visual Phrase Detection (VPDet):** Given an image, detect all relation triplets that exist and assign them a single bounding box that covers the entire “action”. This was first popularised by [27], however given the reliance on full object detectors by most models intended for generating scene graphs, we decided not to consider this task as it is a trivial variation of the more general scene graph generation task.

- **Scene Graph Generation (SGGen):** Given an image, detect all objects in it with bounding boxes and class information, and also all relation triplets between them in order to form a Scene Graph. This is the main task that a full model (incorporating an object detector) should aim to solve. Given the fact that the additional object detection phase introduces a new layer of noise and uncertainty, and in order to produce fair comparisons every model needs to be using the same object detector (which may or may not be possible), we also decided not to consider this task.

Much has been written about evaluation metrics for scene graphs. The most widely used metric is Recall @ K (R@K), which, as explained by [38,57], has problems rooted in the heavily imbalanced distribution of relation classes in datasets such as VG. For this reason, alternatives such as the Zero-shot Recall @ K (zR@K) or Mean Recall @ K (mR@K) metric have also been proposed.

In general, these metrics operate on an image by image basis, and they involve ranking relation triplet predictions by their confidence score generated by the network, and calculating the percentage of a set of ground truth relation triplets that is covered by the top K selection of predicted relation triplets. The metric for a given dataset is calculated by averaging the metrics calculated on every suitable image in the dataset. Recall was chosen as the base metric (as opposed to accuracy) because of the incomplete/inexhaustive nature of the datasets used [27]. Annotations in scene graph datasets do not exhaustively describe every single object and every single relation between them. Using accuracy would result in unfairly penalizing the network for possibly discovering new information about the image that might have been missed by human labellers. For this reason, the problem is approached as an information retrieval or “search” problem, where the goal is returning relevant search results in response to a query.

The following is a summary of how each of these metrics work:

- **Recall @ K (R@K):** This is the base metric that calculates recall over all relation triplets in the ground truth. There is an extra implicit decision affecting the metric, which concerns how many highest scoring predicates to select for each object pair. Some authors consider picking the highest scoring predicate as the only one assigned to a given object pair [27,48], other authors [38,57] decided to select all scores for all predicates, while some others [47,44] decided to make this an explicitly tunable graph constraint hyperparameter k (lowercase, not to be confused with K). This hyperparameter is defined as the number of highest scoring predicates to select from each object pair. Given the multilabel nature of our problem, we decided to follow this last approach and explicitly report which different values are used for both K and k.

- **Zero-shot Recall @ K (zR@K):** This metric evaluates the network’s ability to generalize its understanding of each predicate class by only evaluating the recall on the set of ground truth triplets involving object classes that have not appeared with corresponding predicates in the training set. As an example, (person, laying in, bed) might appear in the training set, but (cat, laying in, bed) might not. zR@K will ignore the former, but consider the latter as part of the ground truth set.

- **Mean Recall @ K (mR@K):** This metric is an attempt to solve the class imbalance problem by calculating R@K independently on each predicate class. In other words, the metric is subdivided into as many metrics as there are predicates. During the final aggregation step over the entire dataset, the individual R@K values of each image are aggregated separately for each predicate (note that the number of values in each group might be different, as some images might not contain examples of certain predicates, and thus they are not considered when calculating the R@K for said groups). The final mR@K value is the arithmetic mean of all R@K values calculated individually for each and every predicate.

In addition, we found that edge cases can arise during the calculation of these metrics in certain situations, the handling of which we believe to be important to fully disclose in order to enable fair comparisons between results.

- Sometimes, images have an empty ground truth triplet set. This can happen because the dataset simply does not record any relation triplets for a given image, or because there are no unseen triplet combinations during zero-shot metric calculation, or because a certain predicate does not appear in the image. We decided to simply skip the image during performance evaluation, since it is not possible to assign a metric to it, as the recall would involve a division by zero.

- The chosen K parameter could be lower than the number of triplets in the ground truth set of some image. In practice this should not happen with the sparse datasets we have, as they contain a low number of triplets per image. Nevertheless, we considered two possible solutions:
  - Imposing a constraint on the value of K so that K is equal or greater than the size of the smallest non-zero
ground truth set.
- Taking the minimum between $K$ and the size of the ground truth set as the divisor when calculating recall. This is the solution we chose, because it does not make sense to calculate a metric in such way that full performance cannot be obtained.

- Some object pairs in the ground truth set might have more predicates than the graph constraint hyperparameter $k$. We did not run into this situation because the test sets we used only have at most a single predicate in each labelled object pair. Nonetheless, we considered a solution, which is to calculate the size of the ground truth set in such way that no more than the given $k$ are considered as the number of predicates accounted for evaluation within each labelled object pair.

- Some authors calculate the recall over the entire dataset instead of averaging the recall values of individual images. This has the effect of slightly underestimating the performance of the model, by giving greater weight to the contribution of images with a larger number of ground truth annotations. This was first notably done by [27], and followed by all papers that compare themselves against [27]. Subsequent works focused on other datasets [38] opt for the more traditional way of aggregating values. We decided to follow existing practice to ensure fairness between results.

- Some authors only calculate recall over the set of object pairs that have corresponding label(s) in the ground truth, instead of taking the scores in all possible object pairs. This has the effect of inflating the reported recall values. This is most notably done when evaluating on the VRD dataset, for consistency with [27].

6. TERESA dataset experiments – Evaluating OG-SGG’s transfer learning capabilities

In order to evaluate the effects of our ontology-guided scene graph generation (OG-SGG) framework, we applied it to a telepresence robotics use case. Specifically, we utilized data from the TERESA [36] European Project, which involved a telepresence robot being used within an elderly day-care centre. The robot is used by both residents and caregivers in order to remotely connect and interact with other people in the centre’s cafeteria (see Figs. 1 or 2 for some examples). We decided to sample a small test set of 25 images from the data recorded by our robot during the project. Afterwards we created a simple ontology using Protégé [29], and annotated all objects on the images with bounding boxes, as well as the corresponding relation triplets. This ontology, although fairly simple, nevertheless encompasses all the important objects and relations of the application scenario. The scheme of the ontology can be seen in Figure 5. The ontology was validated using the FaCT++ [40] reasoner. In order to enable the reproducibility of our research, we published the entirety of the source code we developed, along with our TERESA dataset [1].

The end goal of this experiment is determining whether our techniques allow us to transfer learned knowledge from existing datasets into a completely new problem domain, with minimal work put into defining the rules required to perform this conversion. We also aim to achieve the double-sided benefit of refining the VQA tasks that allow robotic agents to automatically reason about user input, e.g. users of the telepresence system may want to ask about the locations of objects or people, or refer to them by their relation to other entities in the scene.

We also performed an ablation study of our techniques, for the purpose of which we prepared six different training splits for the network. In each split, we tried combinations of input datasets as well as enabling or disabling parts of our dataset filtering/augmentation logic (note that the augmentation logic needs predicates to be already adapted to the ontology, that is, the filtering logic needs to be done previously). We tested VG-SGG’s training split, as well as a filtered version of it only containing images classified as ‘indoors’ by GQA, which we called VG-indoor. This filtered subset, after being processed with our ontology guided procedure, contains around 2500 training images (out of which around 250 are reserved for validation and hyperparameter tuning).

Table 2 shows several statistics about the different dataset splits used for training, as well as our custom TERESA test set used for evaluation. We report the average number of objects per image that are connected by relation triplets, which naturally decreases the more filtering is done. On the other hand, the average number of relation triplets increases even prior to applying the ontology guided augmentation process. We hypothesize this to be caused by the removal of especially noisy or underlabelled images, itself a side effect of the ontology guided filter. The augmentation process causes a milder increase than expected, probably due to the relative simplicity of the ontological model we designed for TERESA. Similar things can also be said about the other metrics, such as the average number of object pairs that are annotated with predicates, or the average percentage of such pairs in the image. This last metric is intended to measure the degree of annotation density (or rather, sparsity) in the triplet annotations by taking the set of objects connected by relation triplets, and expressing the number of pairs with annotations as a percentage of the total number of possible pairs within the set.

6.1. Quantitative results

Table 3 reports evaluation results on the TERESA test set for the network trained on each dataset split, as well as with and without our ontology founded post-processing procedure. We also emphasize that no images from TERESA were used during training. The metrics for the baseline splits were computed by first adapting the output of the model to fit the predicates in the ontology with minimal post-processing and no ontological filtering. Specifically, each predicate in the ontology was assigned an output score of the average of the predicate scores corresponding to its mapped set of predicates (the same used during dataset filtering). We report $R@K$ met-
Figure 5: TERESA ontology, reflecting the dataset’s main entities and relationships among them. The class hierarchy distinguishes GrabbableObject and Furniture in order to separate manipulable objects from pre-existing fixtures that form part of a room visited by the robot, while Person, the main focus of our application, is the domain of certain special purpose predicates such as holding or sitting at/on.

| VG-SGG (Train) | VG-indoor (Train) | TERESA (Test) |
|----------------|-------------------|---------------|
|                | Base | Filter | F.+Aug. | Base | Filter | F.+Aug. |              |
| Number of images | 62723 | 51162 | 51162 | 3036 | 2505 | 2505 | 25 |
| Connected objects/image | 6.73 | 4.59 | 4.60 | 6.16 | 4.31 | 4.31 | 10.42 |
| Triplets/image | 5.46 | 5.91 | 6.02 | 4.72 | 5.27 | 5.33 | 21.46 |
| Annotated pairs/image | 5.15 | 5.79 | 5.89 | 4.48 | 5.19 | 5.25 | 21.38 |
| % pairs with annotations | 9.11 | 22.68 | 23.04 | 9.18 | 24.61 | 24.90 | 18.05 |

Table 2: Dataset statistics. Training datasets are reported in their original base form, their (e.g., domain-) filtered, and their filtered + (ontological axioms-) augmented form.

| Metrics for Predicate Detection (PredDet) |
|------------------------------------------|
| Dataset | R@K (k = 1) | R@K (k = 8) | mR@K (k = 1) | mR@K (k = 8) |
|---------|-------------|-------------|--------------|--------------|
|        | 20 | 50 | 100 | 20 | 50 | 100 | 20 | 50 | 100 | 20 | 50 | 100 |
| VG-SGG baseline | 27.0 | 34.7 | 41.9 | 23.8 | 34.8 | 51.1 | 19.1 | 30.6 | 36.4 | 29.3 | 42.7 | 57.0 |
| VG-SGG with post-processing | 28.4 | 36.0 | 43.2 | 29.5 | 42.2 | 57.9 | 32.4 | 44.6 | 51.1 | 33.5 | 49.7 | 63.0 |
| VG-SGG with filter only | 40.0 | 43.4 | 47.7 | 42.0 | 49.0 | 60.0 | 42.2 | 48.8 | 50.9 | 43.5 | 51.5 | 57.4 |
| VG-SGG with filter + post proc. | 44.7 | 47.9 | 53.2 | **46.5** | 53.4 | 66.3 | 44.0 | 51.2 | 53.6 | 44.7 | 53.9 | 60.5 |
| VG-SGG w/ filter + augmentation | 39.8 | 43.9 | 48.6 | 40.7 | 49.8 | 61.5 | 42.3 | 47.1 | 50.6 | 43.4 | 51.2 | 57.9 |
| VG-SGG w/ f. + aug. + post proc. | **44.9** | **49.3** | **54.0** | **46.5** | **54.0** | **68.3** | 44.6 | 49.9 | 53.3 | 45.2 | 53.0 | 61.7 |
| VG-indoor baseline | 26.1 | 33.7 | 40.2 | 26.0 | 41.4 | 56.1 | 10.5 | 20.6 | 25.2 | 19.8 | 35.5 | 53.6 |
| VG-indoor with post-processing | 30.7 | 38.4 | 45.2 | 32.7 | 45.6 | 59.6 | 22.0 | 39.5 | 44.7 | 23.3 | 39.5 | 58.1 |
| VG-indoor with filter only | 41.2 | 41.3 | 46.7 | 42.2 | 48.1 | 59.2 | 43.7 | 50.6 | 45.5 | 44.7 | 56.0 | 65.8 |
| VG-indoor with filter + post proc. | 43.6 | 45.3 | 51.8 | 44.2 | 51.5 | 62.8 | 45.0 | 52.9 | 59.6 | **47.1** | 57.7 | **69.3** |
| VG-indoor w/ filter + augmentation | 42.1 | 42.6 | 46.9 | 42.2 | 48.5 | 58.1 | 45.5 | 53.0 | 56.1 | 46.2 | 56.8 | 63.8 |
| VG-indoor w/ f. + aug. + post proc. | 44.4 | 46.1 | 51.9 | 43.3 | 51.5 | 62.0 | **46.8** | **54.8** | **61.0** | 46.6 | **58.3** | 66.6 |

Table 3: Evaluation results on TERESA test set. The model was trained on 6 different dataset splits (3 for each baseline dataset), and evaluated with and without post-processing. The different splits are intended to test the efficacy of the filtering and augmentation techniques proposed in this paper.

It must be mentioned that in reality it operates like $zR@K$, as the training and test datasets differ, and so does the set of object classes used in the examples. It can be readily observed that using a training dataset specifically prepared to target a desired set of predicates uplifts performance.

6.1.1. Effects of the baseline dataset

The choice of baseline dataset also produced interesting results. The results for VG-indoor trained models are highly competitive against those trained on regular VG-SGG, despite having over twenty times less images. In some metrics such
Table 4: Qualitative results on TERESA test set.

Top row: Image with object annotations
Middle row: “VG-SGG baseline”
Bottom row: “VG-SGG with filter + augmentation + post processing”

as mR@K it even outperforms VG-SGG, with the full stack version of its model taking the performance crown overall. On the other hand, full stack VG-SGG dominates in R@K. This might be caused by the previously explained problem caused by
predicate bias in R@K. Specifically, mR@K attempts to paint a more balanced picture by weighing the importance of all predicates equally in its formula, therefore allowing the tails of the predicate frequency distribution to have a fair say in the result. Thus, it could be said that VG-indoor was able to generalize better on the less frequent predicates.

6.1.2. Ablation study

**Post-processing.** The ablation study reveals the great importance of the post-processing stage, which enforces the axioms defined in the ontology, purging lower-ranking inconsistent triplet proposals, and thus results in increased performance across the board. An interesting observation can be made for the improvements obtained in baseline models (i.e. when no other components of OG-SGG are used), which bring the metrics close to those observed in filtered datasets (with no post-processing stage). We, as a result, believe this component to be our most significant contribution.

**Filtering.** As previously mentioned, filtering the dataset with the ontology is also majorly responsible for the improved performance. In other words, this process optimizes the transfer learning capabilities of scene graph generation networks, allowing users to obtain better results by “recycling” existing datasets. Specifically, it can be seen that filtering brings significant boosts to mR@K, which is indicative of greater generalization capability. R@K also receives a boost, although it is not as dramatic in comparison.

**Augmentation.** On the other hand, the extra training data produced by the augmentation does not seem to have produced a significant improvement as hoped, i.e. it could be said to lie within the margin of error caused by the variability of the training or stratification processes. This could be caused by not enough complexity/richness in the definition of the TERESA ontology. Nonetheless, we still consider it relevant to continue researching more robust ways of augmenting existing datasets using ontological reasoning.

6.2. Qualitative results

Table shows two selected qualitative examples. The graphs were generated by running the images through the model and picking the 16 highest scoring generated triplets. In the case of the version with post-processing, adding a triplet also adds all associated implicit triplets. Additionally, disallowed triplets, as well as triplets that were previously added implicitly, are not considered in the score ranking, meaning they do not count towards the triplet limit. In the baseline generated graphs, some violations can be spotted (such as multiple people sitting on multiple chairs, or windows sitting at tables). On the other hand, the graphs generated with our proposed techniques have some discernible structures, such as people holding objects, or chairs being next to tables.

With this said, some shortcomings can be seen, such as the generator being unable to tell if an object is being held by a person or is located “on top” of a certain table. In these situations, the generator simply asserts both of these possibilities. In addition, some potentially useful information such as proximity relations between people seems to be deemphasized in favor of other structures that the network was able to learn with the same predicates. Likewise, the network fails to learn cues for discerning the various levels of depth present in images, resulting in the understanding of proximity relationships being reduced to mere 2D spatial proximity, as can be seen in how objects close to people’s hands are nearly always detected as being held. This indicates that more input (such as Depth information) might be necessary for the network, and that higher order ontological rules need to be implemented in order to decide between different possibilities. Nonetheless, there is still a noticeable improvement compared to baseline non-ontology-guided methods.

Sample textual representations of each scene graph revolving around detected people were also generated by enumerating all triplets that have Person entities as their subject. These representations are intended to be an example of downstream automatic scene captioning tasks that are common in telepresence robotics.

7. Conclusion and future work

In this paper we joined the world of ontologies together with the world of scene graph generation, and showed how even the basic strategies we proposed (using only simple filtering and processing based on the most common OWL axioms affecting predicates) can achieve quantitative and qualitative improvements in domain specific environments. While existing scene graph generation networks (such as VRD-RAND) generate all possible pairs, OG-SGG is able to leverage the ontology to reduce the set of possibilities and thus improve the quality of the generated scene graphs. We can observe improvements across the board in R@K, and interestingly enough, training the model with a smaller version of the dataset that only contains images previously known to be relevant to the problem resulted in improved mR@K. Removing graph constraint priors (i.e. by setting the graph constraint hyperparameter k to its highest allowed value) also produced better results.

Another important observation is that only a small amount of effort had to be spent in engineering an ontology for the experiment in order to obtain these results. Specifically, the only two things that need to be done for OG-SGG to work are designing an ontology for the desired problem, and mapping the predicates of the original scene graph dataset to the ones in the ontology. It can be explained that OG-SGG leverages the effect that biased datasets have on neural networks, precisely by creating a new version of the dataset that is biased in favor of existing prior knowledge. On the other side of the equation, OG-SGG also removes outputs that can be safely discarded using the aforementioned prior knowledge. This contrasts with the traditional methods used for transfer learning in neural networks, which are primarily based on hyperparameter tweaking, freezing and unfreezing the weights of individual layers, and other “black box” architectural changes. All of these methods, as with other non-XAI techniques, cannot be driven by human intuition or by pre-existing knowledge; and as such take a considerably higher amount of effort to refine, mostly through pure trial and error.

Nevertheless, there is margin for further refinements and fil-
tering. It still takes a high K cutoff to capture a sizable majority of the triplets present in ground truth annotations, indicating a need for better filtering. The quality of the filtering also depends on how detailed the ontology is—naturally, the more axioms and predicates that exist the more precise the predictions will be. In addition, a major flaw with existing scene graph generation networks can be observed, which is the difficulty of defining a score threshold for dropping unlikely relation triplets. Currently, a basic Top-K strategy is still used, which tends to leave out perfectly valid predictions in crowded scenes. For this reason, a possible future direction would be to design a new post processing stage based on a neural network that draws the line for us, and possibly even go further by filtering with a score function from the ontology. Another area of interest for possible future research is integrating existing ontological knowledge directly into the main scene graph generation network, perhaps in the form of a new term in the loss function [14], or through incorporating neurosymbolic propositional and first order logic directly as part of the training process [9].

On an ending note, we propose further research on downstream usages of OG-SSG such as knowledge-graph driven image captioning or robotic visual question answering, further leveraging structured approaches to incorporating prior relevant knowledge.

Acknowledgment

This work is partially supported by Programa Operativo FEDER Andalucía 2014-2020, Consejería de Economía y Conocimiento (TELEPORTA, UPO-1264631) and the project PLEC2021-007868, funded by MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/PRTR. N. Díaz-Rodríguez is supported by the Spanish Government Juan de la Cierva Incorporación contract (JCI2019-039152-I) and Google Research Scholar Programme.

References

[1] Pranav Agarwal, Alejandro Betancourt, Vana Panagiotou, and Natalia Díaz-Rodríguez. Egoshots, an ego-vision life-logging dataset and semantic fidelity metric to evaluate diversity in image captioning models. arXiv preprint arXiv:2003.11743, 2020.
[2] Alexandre Armand, David Filliat, and Javier Ibáñez-Guzmán. Ontology-based context awareness for driving assistance systems. In 2014 IEEE intelligent vehicles symposium proceedings, pages 227–233. IEEE, 2014.
[3] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrién Bennettot, Siham Tabik, Alberto Barbudo, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamin, et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58:82–115, 2020.
[4] Samy Badreddine, Arnt d’Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks, 2022.
[5] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.
[6] Guilhem Buisan, Guillaume Sarrith, Arthur Bit-Monnot, Aurélie Clodic, and Rachid Alami. Efficient, Situated, and Ontology based Referring Expression Generation for Human-Robot collaboration. In 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 349–356, 2020.
[7] Ferenc Bálint-Benczédi, Jan-Hendrik Worch, Daniel Nygå, Nico Böladow, Patrick Mania, Z. Márton, and Michael Beetz. RoboSherlock: Cognition-enabled Robot Perception for Everyday Manipulation Tasks. ArXiv, abs/1911.10079, 2019.
[8] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Loddon Vuille. Detect What You Can: Detecting and Representing Objects Using Holistic Models and Body Parts. 2014 IEEE Conference on Computer Vision and Pattern Recognition, pages 1979–1986, 2014.
[9] Devleena Das, Siddhartha Banerjee, and Sonia Chernova. Explainable AI for Robot Failures: Generating Explanations that Improve User Assistance in Fault Recovery. In RRI ’21: ACM/IEEE International Conference on Human-Robot Interaction, Boulder, CO, USA, March 8-11, 2021, pages 351–360, 03 2021.
[10] Devleena Das and Sonia Chernova. Semantic-Based Explainable AI: Leveraging Semantic Scene Graphs and Pairwise Ranking to Explain Robot Failures. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2021, Prague, Czech Republic, September 27 – Oct. 1, 2021, pages 3034–3041, 09 2021.
[11] 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.
[12] Ivan Donadello and Luciano Serafini. Integration of numeric and symbolic information for semantic image interpretation. Intelligenza Artificiale, 10(1):33–47, 2016.
[13] Natalia Díaz-Rodríguez et al. Semantic and fuzzy modelling of human behaviour recognition in smart spaces. A case study on ambiental assisted living. PhD thesis, Universidad de Granada, 2015.
[14] Natalia Díaz-Rodríguez, Alberto Lamas, Jules Sanchez, Gianni Franchi, Ivan Donadello, Siham Tabik, David Filliat, Polacipo Cruz, Rosana Montes, and Francisco Herrera. Explainable Neural-Symbolic Learning (X-NeSyL) methodology to fuse deep learning representations with expert knowledge graphs: The MonuMAI cultural heritage use case. Information Fusion, 79:58–83, 2022.
[15] Nicola Guarino, David Obrèbè, and Steffen Staab. What Is an Ontology?, pages 1–17. Springer, 05 2009.
[16] Drew A Hudson and Christopher D Manning. GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering. Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
[17] IEEE. IEEE Standard Ontologies for Robotics and Automation. IEEE Std 1872-2015, pages 1–60, 2015.
[18] 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.
[19] Petteri Karvinen, Natalia Diaz-Rodriguez, Stefan Grönnroos, and Johan Liljus. RDF stores for enhanced living environments: an overview. Enhanced Living Environments, pages 19–52, 2019.
[20] Rajat Koner, Poulam Sinhamahapatra, and Volker Tresp. Relation transformer network. arXiv preprint arXiv:2004.06193, 2020.
[21] 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(1):32–73, 2017.
[22] Anna Kristoffersson, Silvia Coradeschi, and Amy Loutfi. A Review of Mobile Robotic Telepresence. Adv. in Hum.-Comp. Int., 2013:3–3, January 2013.
[23] Jean-Baptiste Lamy. Owready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. Artificial Intelligence in Medicine, 80:11–28, 2017.
[24] Kongming Liang, Yuhong Guo, Hong Chang, and Xilin Chen. Visual Relationship Detection with Deep Structural Ranking. In AAAI Conference on Artificial Intelligence, 2018.
[25] Y. Liang, Y. Bai, W. Zhang, X. Qian, L. Zhu, and T. Mei. ViR/VG: Refocusing Visually-Related Relationships. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 10402–10411. Los Alamitos, CA, USA, nov 2019. IEEE Computer Society.
[26] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David Fleet, Tomas Pajda, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision — ECCV 2014, pages 745–755, Cham, 2014. Springer International Publishing.
[27] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual Relationship Detection with Language Prior. In European Conference
Tell: Neural Image Caption Generation with Visual Attention. In Francis Bach and David Blei, editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 2048–2057, Lille, France, 07–09 Jul 2015. PMLR.