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]

[1]Service Robotics Laboratory, Universidad Pablo de Olavide, Seville, Spain (e-mail: famozur@upo.es; lmercab@upo.es)
[2]Service Robotics Laboratory, Universidad de Sevilla, Spain (e-mail: fcaballero@us.es)
[3]Computer Science and Artificial Intelligence Dept., Andalusian Research Institute in Data Science and Computational Intelligence (DaSCI), University of Granada, Spain (e-mail: nataliadiaz@ugr.es)

Corresponding author: Fernando Amodeo (e-mail: famozur@upo.es)

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.

INDEX TERMS Scene Graph Generation, Ontology, Computer Vision, Telepresence Robotics

I. 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 [26]. 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, this work considers the application of telepresence robots for elderly care [42] (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

FIGURE 1. A telepresence robot for elderly care. The robot in this picture is used to remotely assist with the activities of a day care centre.

that is given to the high-level tasks [47]. 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 one considers 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.

This work considers recent advances in visual scene graph generation (i.e. [50, 43, 24, 54]) 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. For example, it is easy and intuitive for a human to understand that a `Person` can only be sitting on a single `Chair` at a given time. These current solutions have no means of providing common-sense axioms that can help filter out inconsistent detections, and thus they can erroneously predict that a person is sitting on multiple chairs at the same time. These axioms can be collected using what is known as an *ontology*, and it is the main pillar that enables this work.

The main goal of this 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. In particular, this work proposes the following contributions:

- A methodology for augmenting an existing scene graph generator with the addition of steps involving ontology-founded reasoning, as opposed to simply defining a new model.
- A recipe to prepare an existing dataset for a desired application by applying an ontology, filtering irrelevant information and including inferred knowledge as a form of data augmentation.

The approach is validated in a real transfer learning case for ontology-based generation, considering a small specialized dataset and ontology founded in a robotics application.

II. BACKGROUND
Ontologies are broad constructs that can be used to represent the cognitive model of a given domain world [18]. In simplified terms, 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 [20, 35] and perform context-awareness reasoning [2, 16]. Ontologies can also be used to model knowledge graphs with richer and more formal semantics, allowing for higher order reasoning.

This work is based on OWL 2 [48], the standard knowledge representation language for defining ontologies created by the World Wide Web Consortium (W3C). OWL is built upon RDF [22], 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 [3] in the form of knowledge graphs and other semantic web technologies [40].

Given a scene and an ontology, it is possible to 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 $P_{ij} = \{p_k \mid (o_i, p_k, o_j) \in R\}$, 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 object pairs (o_i, o_j). 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. This is the approach considered in this work.

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; whereas 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 and the rest being discarded.

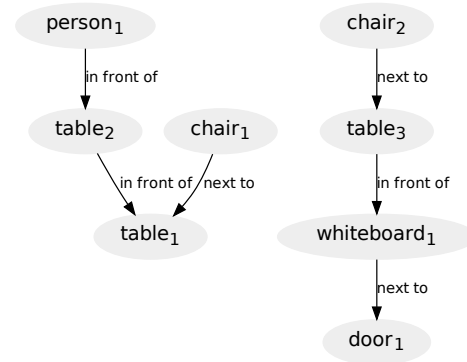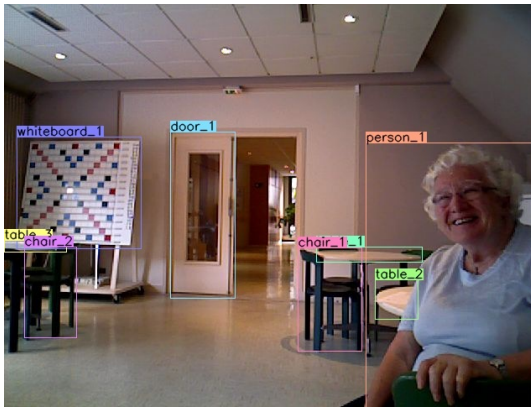This work proposes a series of techniques to improve this

FIGURE 2. An image captured by the robot, manually annotated with a scene graph, part of the TERESA dataset.

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. Specifically, the following types of axioms that affect predicates, which are defined by [48], have been considered:

- 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 ($plant_1$, `sitting on`, $food_1$).
- 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 ($person_1$, `behind`, $chair_1$) and ($chair_1$, `behind`, $table_1$) hold, then ($person_1$, `behind`, $table_1$) 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 ($person_1$, `holding`, $pencil_1$) holds, then no other `Person` object can be related to $pencil_1$ 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, ($chair_1$, `next to`, $table_1$) implies ($table_1$, `next to`, $chair_1$).

III. RELATED WORK

Given the previously mentioned use case in telepresence robotics, we initially surveyed simpler, more direct approaches such as automatic image captioning [52], 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 was confirmed by other researchers using indicators such as Semantic Fidelity [1]), 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 shift our attention towards scene graphs, specifically existing work concerning their automatic generation. This section contains a detailed 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.

A. METHODOLOGIES

A line of research that we deemed relevant revolves around neurosymbolic (NeSy) computation approaches. This interest arose from our need to improve the specialized application of a deep learning model towards a certain application. Several approaches exist, including ones that are oriented towards improving the explainability of AI solutions (XAI) [17, 5], an increasingly important topic.

Some approaches such as [15] propose applying NeSy logic constraints in the form of Logic Tensor Networks [4] to scene graph generation. These approaches map discrete logical operators to continuous differentiable operators based around fuzzy logic. However, they only consider the traditional approach of using training and testing splits of the same dataset, leaving out the desired goal of transferring knowledge learned from one dataset to a different scenario. Moreover, a new model architecture specifically designed to incorporate NeSy components had to be created, which might not always be feasible to implement depending on the constraints of the intended application.

B. 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 the domain-specific scene graph generation task:

- MS COCO [30] 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 [14] is an OWL conversion of an earlier PASCAL-Part dataset [9]. 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 [32] (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 [21], which was itself sampled from the intersection of MS COCO and YFCC100m [45].

- VG [25] (Visual Genome)[1] 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 [33] to identify objects and relations, which adds a 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 [51] 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 [29] (Visual-relevance Relations) is another filtered and improved version of VG specifically in-

tended 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 [19] (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 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.

This work proposes 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.

C. 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 [32] 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 [51] 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 [54] introduced a new architecture based on bidirectional LSTMs that was capable of detecting patterns in the structure of the scene graphs called "motifs."

---

[1] http://visualgenome.org/

IEEE *Access*

- VRD-DSR [28] 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 [43] 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 [41] 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 [50] 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 [24] 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-RCNN [38] (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, nor evaluations of individual components. These factors make it needlessly difficult to adapt existing scene graph generation solutions to new problem domains such as the previously mentioned robotics application.

Another problem we identified with existing solutions is that they are designed as pure deep learning architectures and, as such, they cannot take advantage of subtle semantics concerning the defined object and relation classes that are implicit and intuitive for humans. Thus, it is common for SGG models to output scene graphs full of inconsistencies. The methodology proposed in this work is capable of improving the usability of scene graph generation for specific domains, and is independent of the model used. That is, it 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).

D. ROBOTICS RELATED RESEARCH
Several works with some relevance to this problem have previously been published, including the following:

- Ontologenius [39] is a semantic memory module based on OWL ontologies for the ROS [36] 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 [7] make use of it as the underlying knowledge engine.
- RoboSherlock [8] 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 [10, 11] 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 usage 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. Even though this work showcases a specific application, it aims to provide methods that can be reused in other applications without significantly changing how they work.

IV. METHODS
The 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 this work's contribution, make use of pre-existing expert knowledge defined in the domain ontology, whereas the network itself can be adapted from the existing state of the art with minimal changes according to needs (such as efficiency).

A. SCENE GRAPH GENERATION NETWORK
OG-SGG augments an existing scene graph generation network. The only requirements imposed by our proposed methodology on this SGG network are the following:

- It needs to be able to detect and process objects within the image, either as part of a combined object detection–relationship detection network, or as a standalone component that reuses the output of an existing object detector. In later sections of this paper, we focus on the latter case in order to evaluate our methodology purely on the quality of the detected relationships, and not the object detection.
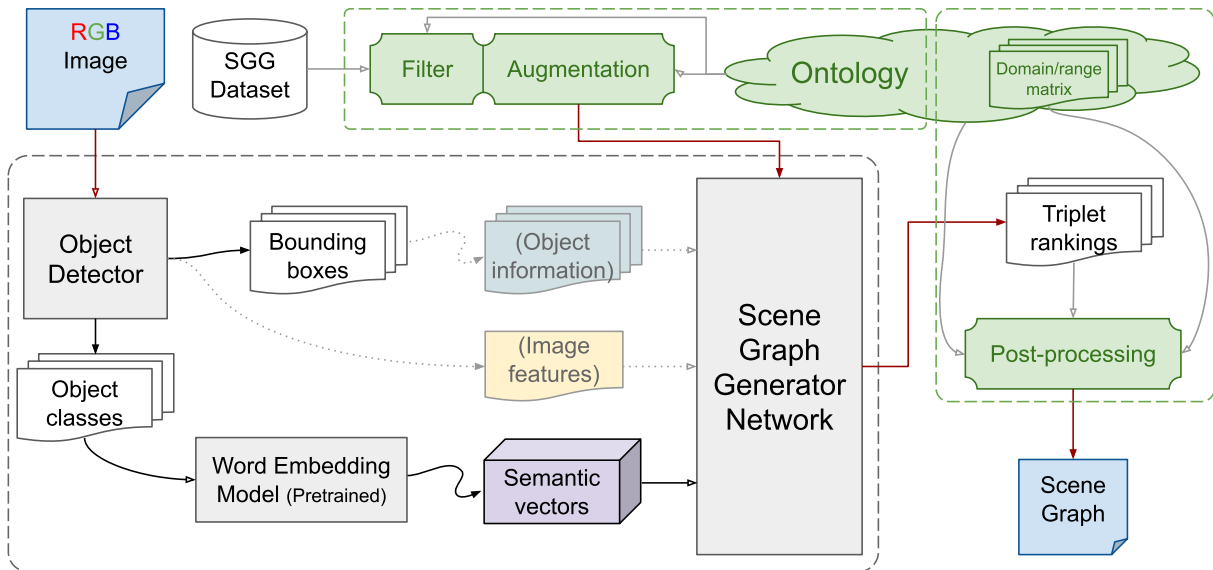
FIGURE 3. Full OG-SGG pipeline proposed protocol. The diagram shows its three main components in full detail, along with both internal and external data flows. The proposed ontology-aware additions are highlighted in green, which include the filter/data augmentation component, and the post-processing component. A pre-existing object detection and scene graph generation component is also included in the pipeline. The latter receives semantic vectors as input, and optionally other data such as object location information and visual feature maps.

- It needs to receive semantic information about the types of objects within the scene in the form of dense semantic vectors, as opposed to other ways such as one-hot encodings. 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 enable the generalization capability of the network, and is in fact required so that the network can be repurposed for different sets of possible input objects without needing retraining.
- It needs to output a single ranking value for every possible knowledge triplet proposal, across all detected objects and proposed relationship types – no range restrictions are applied.

Once all triplet ranking scores are calculated, they are sorted in descending order. The threshold below which triplet proposals are judged as more unlikely than likely is usually left undefined, therefore requiring users of the network to establish their own post-processing rules. A later section of this work will revisit this area, where post-processing rules that take into account information expressed in the ontology will be proposed.

B. DATASET FILTERING AND DATA AUGMENTATION

We decided to focus this 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. 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[2] 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 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 the given problem domain's ontology (see Fig. 4). 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 [27] as the software library providing ontology

[2]https://tfhub.dev/google/Wiki-words-250-with-normalization/2

IEEE *Access*

```
[teresa.predicate_map]

"in front of" = [ "in front of" ]
"next to"     = [ "near" ]
"below"       = [ "under" ]
"behind"      = [ "behind" ]
"on top of"   = [ "above", "laying on", "mounted on", "on", "over" ]
"holding"     = [ "carrying", "holding" ]
"sitting at"  = [ "sitting on" ]
"sitting on"  = [ "sitting on" ]


[ai2thor.predicate_map]

"above"       = [ "above" ]
"below"       = [ ] # inverse of above
"has"         = [ "has" ]
"near"        = [ "near" ]
"lying on"    = [ "lying on" ]
"sitting on"  = [ "sitting on" ]
"standing on" = [ "standing on" ]
"part of"     = [ ] # inverse of has
```

FIGURE 4. Examples of predicate maps. Each key corresponds to a predicate in the ontology, whereas each value is a list of equivalent predicates in the source dataset.

FIGURE 5. Different post-processing techniques proposed in this section. **Left**: Domain/range constraint tensor C. **Right**: Axiom-based pruning.

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, $(chair_1, next\ to, table_1)$ would generate $(table_1, next\ to, chair_1)$. Likewise, $(cup_1, on\ top\ of, table_1)$ would generate $(table_1, below, cup_1)$. 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.

C. 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.
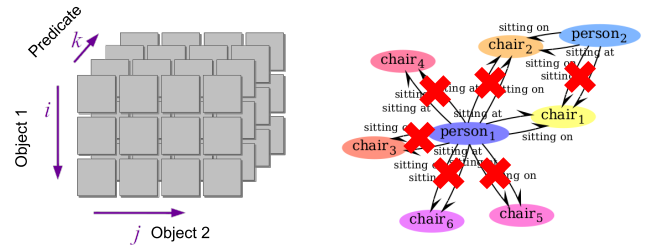
We propose appending a new output post-processing stage to the prediction process that prunes relation triplets that introduce violations of axioms defined in the ontology (see Fig. 5). We studied in Section II the leveraging of several kinds of axioms that affect predicates, such as Functional/InverseFunctional restrictions, or domain/range restrictions. The general filtering approach consists of only accepting the highest ranking mutually exclusive triplet proposals and pruning the rest. For example, if there are two triplet proposals, $(person_1, sitting\ on, chair_1)$ with score 0.78 and $(person_1, sitting\ on, chair_2)$ with score -0.4, the first triplet is accepted and the second one is pruned.

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 $|O| \times |O| \times |P|$ (see Fig. 5), 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).

V. EXPERIMENTAL SETUP

We settled on VRD-RANS [50] as the baseline scene graph generation network for experiments in this work, which was

implemented as faithfully as possible. This network was chosen for the following reasons:

- 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 [37] or SSD [31] 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 [51, 54, 43], 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. Generating a scene graph involves feeding all object pairs to the network in order to extract the ranking scores, which are raw unbounded values $\in \mathbb{R}$ with no defined semantics other than comparison operators (i.e. $<, >, \leq, \geq, =, \neq$).

A. NETWORK LAYERS

The network (illustrated in Figure 6) receives four inputs, three of which are given for each individual (object pair), and the other one is shared across all individuals belonging to a given image. Specifically, the network accepts an object mask (two channels, one for each object in the pair), two semantic vectors, and the global feature map of the image. Objects masks are simple matrices where each pixel falling within the bounding box of a detected object is set to 1, and the rest remain 0. These masks are resized to a fixed dimension in order to improve efficiency.

The first component of the network is the non-visual vector generator. This module fuses all per-object-pair inputs, and outputs a new vector which is later used by the core part of the network. This fusing is performed in two steps: first, the object mask is processed by three convolutional layers followed by a dense layer, and in parallel the two semantic vectors are concatenated and fed to two consecutive dense layers. The outputs from these two branches are finally concatenated in order to form the final non-visual vector.

The core part of the network receives the non-visual vector and the visual feature map as inputs. Before the main loop, a new *visual vector* variable is initialized by performing a 2D global average pool of the entire feature map. Additionally an accumulator vector is initialized with the output of a single dense layer receiving the concatenated visual and non-visual vectors. This dense layer, which is the final layer of the network, has as many outputs as predicates defined in the ontology.

The network performs a recursive process with a fixed number of iterations (5 was used, same as [50]). In each iteration, the concatenated visual and non-visual vectors are processed by two other consecutive dense layers before arriving to a GRU. The GRU's hidden state is zero-initialized, and in each following iteration it will contain the output hidden state from the previous iteration. The GRU's output is then processed by four chained transposed convolutional layers, which in turn generate a new attention mask encoding which locations in the image are to be "looked at" next. The size of this attention mask does not match that of the feature map, so it is necessary to resize it to the same dimensions (using linear interpolation). This new resized attention mask is then used as the weights for a new 2D global average pool of the feature map (from which a single scalar value is obtained from each channel). This new output is stored in the visual vector, and finally the accumulator is updated by adding the new output of the final dense layer to it.

After performing all iterations, the accumulator is divided by one plus the total number of iterations, calculating thus the arithmetic mean of all outputs provided by the final dense layer. This value is the final output of the network.

In general, the activation function used throughout the entire network is ReLU, due to its simplicity and efficiency. In order to improve the stability of the training process, several batch normalization layers were placed between consecutive dense layers. Also worth mentioning is the fact that the attention mask is generated using softmax activation so that it can be used as weights for a weighted mean (in other words, so that all coefficients add up to 1). The final layer of the network does not use any activation function. This is required so that ranking values can be generated.

B. LOSS FUNCTION

As in VRD-RANS [50], 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:

$$\mathcal{L} = \frac{1}{Nn} \sum_{\forall i | y_i = 0} \sum_{\forall j | y_j = 1} \max\left(0, 1 - (\hat{y}_j - \hat{y}_i)\right)$$

where N is the number of object pairs in the minibatch (i.e. its size), and 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), and $\hat{y}_i$ corresponds to the output of the network, that is, the ranking scores $\in (-\infty, +\infty)$
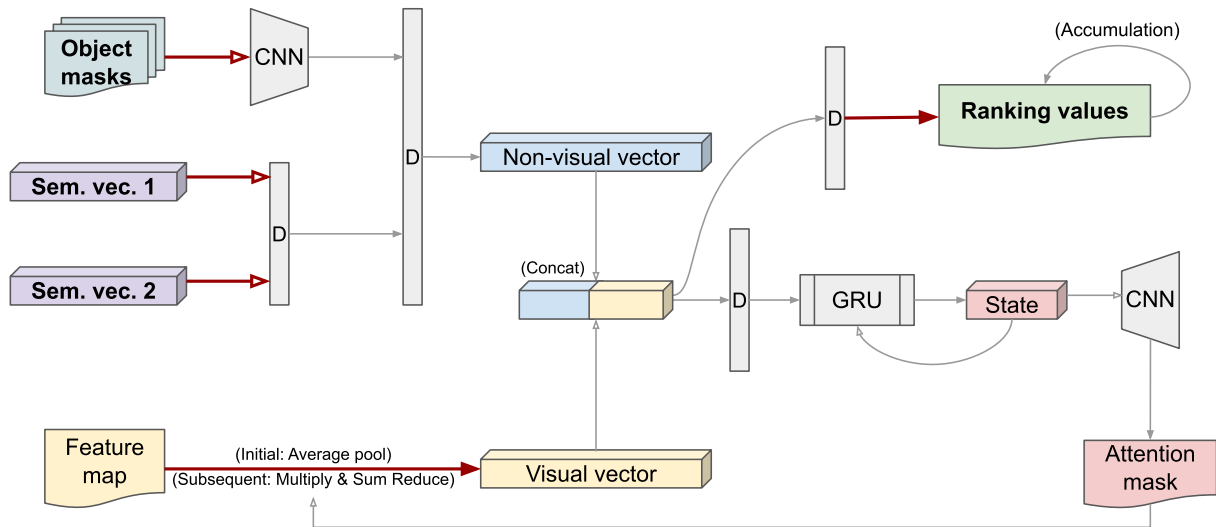
IEEE *Access*

FIGURE 6. Diagram of the overall network structure of VRD-RANS [50]. This is the chosen scene graph generation network for use in our experiments. The network, which is divided into an initial feed-forward part and a main recursive part, contains several Dense layers (marked with a D); two series of CNN layers (one of which is transposed), and a single Gated Recurrent Unit (GRU). The visual vector is initialized with an average pooling of the feature map for the first iteration of the recursive part, and in subsequent iterations the averaging is additionally weighted using the freshly calculated attention mask. The non-visual vector is initialized by the feed-forward part, which takes the semantic vectors and object masks as input, and remains unmodified throughout the execution of the recursive part.

predicted by the network for each triplet. This function, which takes the entire mini-batch 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 ($j \mid y_j = 1$).

C. IMPLEMENTATION DETAILS

TABLE 1. Model hyperparameters used for VRD-RANS

Hyperparameter	Value
Feature map size	$10 \times 10 \times 512$
Object mask size	$64 \times 64 \times 2$
Semantic vector size	250
Optimizer	AdamW
Learning rate	10^{-5}
Weight decay	10^{-5}
Early stopping patience	2 epochs
Batch size	32

We selected YOLOv4 [6] 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 [30], which include a CSPDarknet53 [49] backbone pretrained on ImageNet [13].

The original VRD-RANS [50] authors seemingly did not publish their code. For this reason, this work includes a new implementation of their proposed network. This new implementation was carried out using the TensorFlow framework, with the high-level Keras API layered on top. A new subclass of `tf.keras.Model`, called `RelationshipDetector`, implements the network, and a further subclass of it in turn (called `TelenetTrainer`) implements the special training procedure required by VRD-RANS. Sampling probabilities or priors were not imple-

mented (unlike [50]) in order to maximize the zero-shot perfomance 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 counterproductive, as we are precisely trying to apply the system to a transfer learning problem.

The hyperparameters used in this work are listed in Table 1. The AdamW optimizer was selected instead of the classic Adam because it is capable of performing weight normalization automatically, and thus it is possible to avoid needing to explicitly specify normalization strategies in each layer. The rest of the hyperparameters were found empirically. The network was trained on a single NVIDIA Quadro RTX 5000 GPU.

The training process dynamically generates a minibatch for each image in the training set (`prepare_minibatch` method). The minibatch is generated by separately sampling object pairs with ground truth predicate labels (*positive*) and unlabelled object pairs (*negative*). The same number of positive and negative pairs are sampled. If this is not possible due to there not being enough pairs of a certain kind, the minibatch is filled with pairs of the other kind. If there are simply not enough pairs in total, it is filled with random duplicate copies until the desired minibatch size is reached.

D. 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 [43, 54] 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 [32], 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 [44, 43], 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 [32]. 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 [32, 54], other authors [44, 43] decided to select *all* scores for all predicates, whereas some others [53, 50] 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 this 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 [32], and followed by all papers that compare themselves against [32]. Subsequent works focused on other datasets [44] 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 [32].

VI. TERESA DATASET EXPERIMENTS. EVALUATING OG-SGG'S TRANSFER LEARNING CAPABILITIES

In order to evaluate the effects of the ontology-guided scene graph generation (OG-SGG) framework, we applied it to a telepresence robotics use case. Specifically, we utilized data from the TERESA [42] 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). 13 sessions were carried out in total, during which large amounts of data were collected from the robot's cameras and other sensors. For the purposes of these experiments, a small sample of 25 images were extracted and manually annotated with the objects and relationships present within them.

The end goal of this experiment is determining whether these 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. In order to enable the reproducibility of this work, we published the entirety of the source code we developed, along with the TERESA dataset [3].

We created a simple ontology using Protégé [34], 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 7. The ontology was validated using the FaCT++ [46] reasoner. Several top level *classes* were defined: `Furniture` (stationary objects present in a room), `GrabbableObject` (objects that can be grabbed and moved by the robot) and `Person` (corresponding to humans). A few *object properties* were also defined, corresponding to common relationships between entities in a scene graph. For example, `on top of` is a Functional object property with the domain `Appliance or GrabbableObject` and the range `Appliance or Counter or Table`. This defines several axioms: 1) something can only be on top of a single surface (not several at the same time), 2) only appliances (e.g. microwaves) or small, movable objects can be on top, whereas 3) said acceptable "surfaces" can only be tables, counters or other appliances. Another example is `sitting on`: a Functional and InverseFunctional property with the domain `Person` and the range `Chair`. This enforces that people can only be sitting on a single chair, and a chair cannot host multiple people at the same time.

We also performed an ablation study of these 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 the 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 the ontology guided procedure, contains around 2500 training images (out of which around 250 are reserved for validation and hyperparameter tuning), and is intended to test OG-SGG in cases where the source dataset is considerably smaller in size and scope.

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,
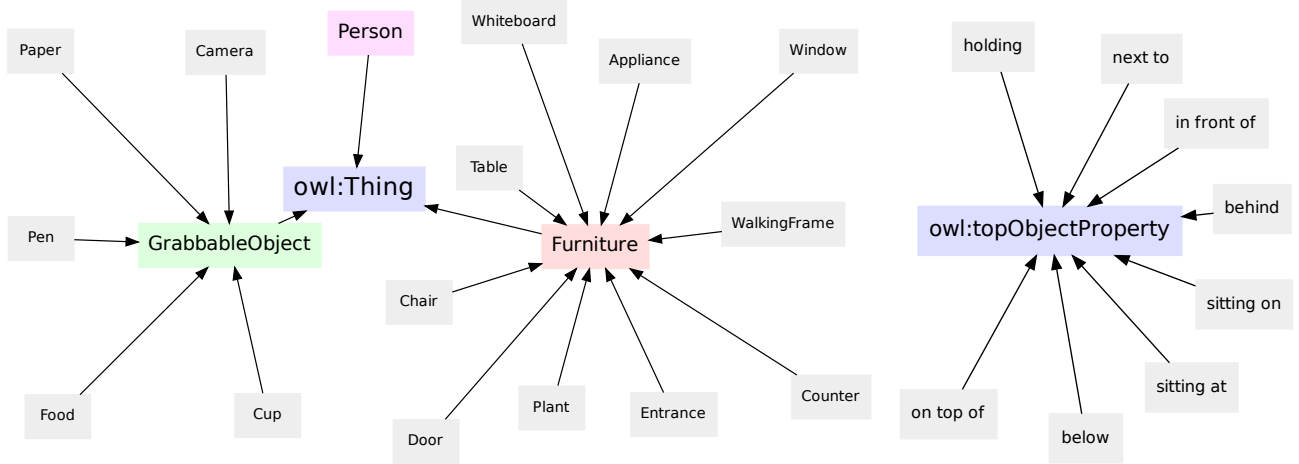
[3]https://github.com/robotics-upo/og-sgg

FIGURE 7. 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, whereas `Person`, the main focus of a telepresence robotics application, is the domain of certain special purpose predicates such as `holding` or `sitting at/on`.

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

	VG-SGG (Train)			VG-indoor (Train)			TERESA
	Base	Filter	F.+Aug.	Base	Filter	F.+Aug.	(Test)
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 3. Evaluation results on TERESA test set. The model was trained on 6 different dataset splits (3 for each source training dataset), and evaluated with and without post-processing (*"Post"* column). The different splits are intended to test the efficacy of the filtering and data augmentation techniques proposed in this work. The different metrics are computed for different top **K** predicted relation triplets (20, 50 and 100) per image and different graph constraint hyperparameter **k** values (1 and 8 predicates per object pair). The best results for each source training dataset are marked in bold.

Dataset	Post	Metrics for Predicate Detection (PredDet)											
		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 unmodified dataset	✗	27.0	34.7	41.9	23.8	34.8	51.1	19.1	30.6	36.4	29.3	42.7	57.0
	✓	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 filtering	✗	40.0	43.4	47.7	42.0	49.0	60.0	42.2	48.8	50.9	43.5	51.5	57.4
	✓	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 with filtering/augmentation	✗	39.8	43.9	48.6	40.7	49.8	61.5	42.5	47.1	50.6	43.4	51.2	57.9
	✓	**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 unmodified dataset	✗	26.1	33.7	40.2	26.0	41.4	56.1	10.5	20.6	25.2	19.8	35.5	53.6
	✓	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 filtering	✗	41.2	41.3	46.7	42.2	48.1	59.2	43.7	50.6	45.5	44.7	56.0	65.8
	✓	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 with filtering/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
	✓	**44.4**	**46.1**	**51.9**	43.3	**51.5**	62.0	**46.8**	**54.8**	**61.0**	46.6	**58.3**	66.6

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 data 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 data 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.

A. 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 the ontology founded output post-processing procedure. We also emphasize that no images from TERESA were
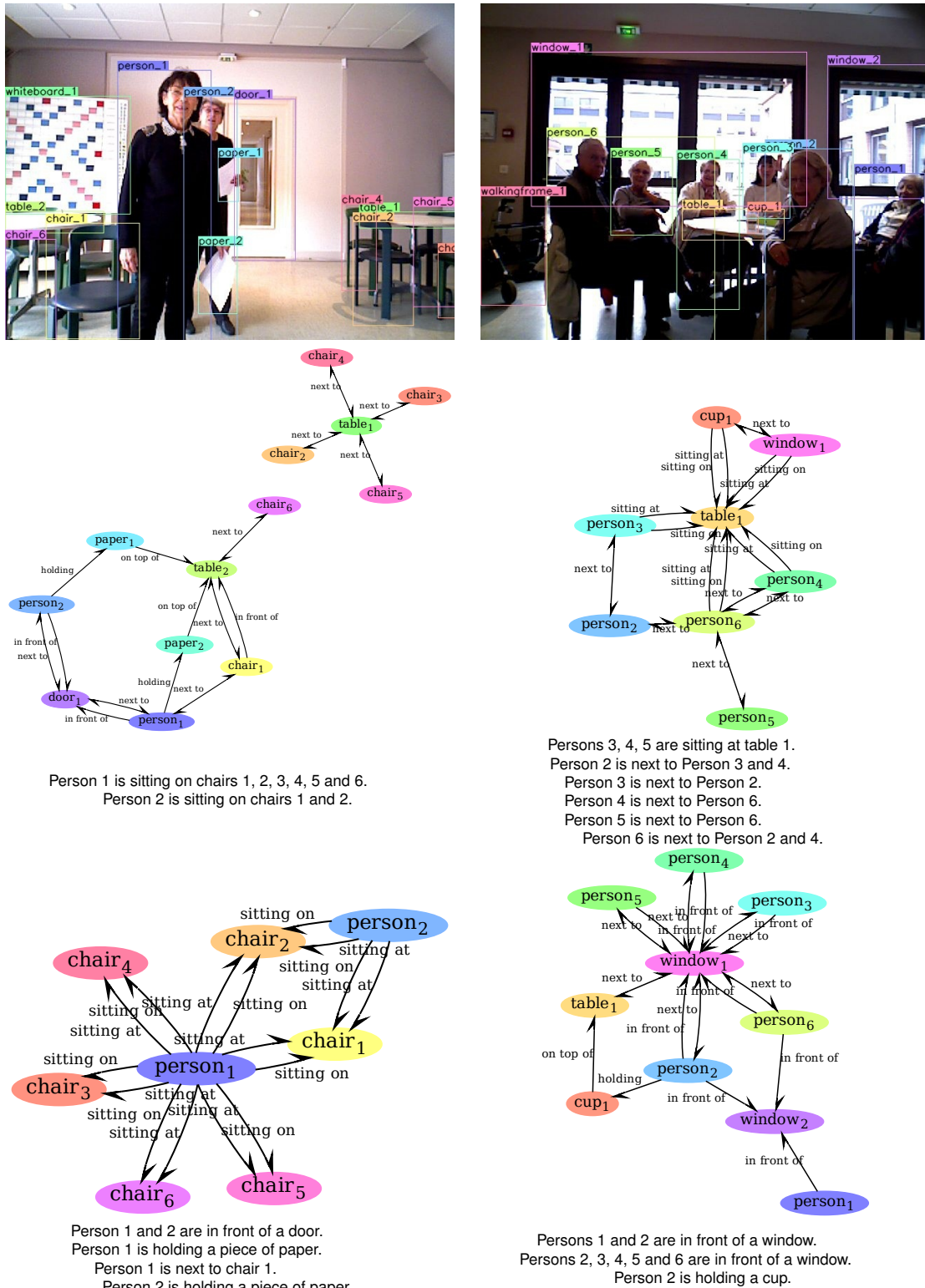
FIGURE 8. Qualitative results on TERESA test set, gathered using the TERESA robot of Fig. 1. **Top row**: Image with object annotations. **Middle row**: "VG-SGG baseline without OG-SGG." **Bottom row**: "VG-SGG with full stack OG-SGG."

used during training. The metrics for the splits corresponding to unmodified source training datasets 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). Although we report **R@K** metrics, the network is trained on a completely different dataset to the one used for testing; and the set of semantic vectors corresponding to object classes seen by the network during training is also entirely different from the one provided during testing – in other words, the metrics are calculated purely on zero-shot triplets never seen during training, thus effectively calculating a sort of **zR@K**. It can be readily observed that using a training dataset specifically prepared to target a desired set of predicates uplifts performance.

1) Choice of training dataset

The choice of original dataset onto which OG-SGG is applied also produced interesting results. The results for VG-indoor trained models are highly competitive against those trained on regular VG-SGG. In some metrics such as **mR@K** it outperforms VG-SGG trained models, with the full stack OG-SGG version of its model taking the performance crown overall. Even more, the model trained on VG-indoor with full stack OG-SGG outperforms the VG-SGG baseline without OG-SGG, despite having seen over twenty times less images during the training process.

On the other hand, models trained on VG-SGG with full stack OG-SGG dominate 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 frequence distribution to have a fair say in the result. Thus, it could be said that training on VG-indoor helped the model generalize better on the less frequent predicates.

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 when training on unmodified datasets (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 the most significant contribution of this work.

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.

B. QUALITATIVE RESULTS

Fig. 8 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 the proposed techniques in this work 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.

VII. ADDITIONAL EXPERIMENTS USING AI2THOR

We decided to apply OG-SGG to a similar but different robotics scenario. Specifically, the AI2THOR framework [23] is a near photo-realistic interactable framework for embodied AI agents, with the goal of facilitating the creation of visually intelligent models and pushing the research forward in that domain. One of the environments present in this framework is RoboTHOR [12] which has a specific focus on

IEEE *Access*

TABLE 4. Dataset statistics for AI2THOR. Training datasets derived from VG are reported in their original base form, their (e.g., domain-) filtered, and their filtered + (ontological axioms-) augmented form.

	VG-SGG (Train)			VG-indoor (Train)			AI2THOR
	Base	Filter	F.+Aug.	Base	Filter	F.+Aug.	(Test)
Number of images	62723	34585	34585	3036	1679	1679	113
Connected objects/image	6.73	3.83	3.84	6.16	3.67	3.67	12.42
Triplets/image	5.46	4.82	4.84	4.72	4.36	4.38	48.43
Annotated pairs/image	5.15	4.78	4.80	4.48	4.31	4.33	45.11
% pairs with annotations	9.11	28.40	28.51	9.18	26.83	26.90	27.27

TABLE 5. Evaluation results on AI2THOR test set. The model was trained on 6 different dataset splits (3 for each source training dataset), and evaluated with and without post-processing (*"Post"* column).

Dataset	Post	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 unmodified dataset	✗	19.1	24.4	32.1	21.9	27.2	40.5	7.0	11.8	16.2	9.4	15.2	24.6
	✓	**21.0**	**29.7**	**40.4**	**23.7**	32.7	48.3	7.5	13.5	18.3	10.1	17.6	28.2
VG-SGG with filtering	✗	16.7	27.5	37.1	17.2	28.0	44.3	11.5	20.6	27.3	13.0	24.4	42.4
	✓	17.3	28.7	38.7	18.9	**33.1**	**52.5**	11.6	21.1	**27.9**	**13.8**	28.0	**48.3**
VG-SGG with filtering/augmentation	✗	19.3	27.7	36.8	19.7	28.3	43.4	12.3	20.5	26.6	13.2	26.0	40.4
	✓	20.0	29.0	38.5	20.9	28.4	50.7	**12.9**	**21.2**	27.2	**13.8**	**29.0**	45.3
VG-indoor unmodified dataset	✗	18.7	21.4	26.0	22.1	26.5	37.4	9.3	17.7	23.0	12.2	22.3	31.5
	✓	**22.6**	**27.8**	**34.5**	**25.3**	31.1	45.1	10.8	20.7	25.7	13.5	24.8	35.8
VG-indoor with filtering	✗	20.5	23.4	28.1	23.2	28.4	39.0	14.0	22.7	28.4	16.0	28.8	42.6
	✓	21.4	25.7	31.4	24.7	**32.9**	**45.9**	**14.5**	**24.1**	**30.8**	**16.9**	**32.5**	**48.0**
VG-indoor with filtering/augmentation	✗	19.3	22.2	26.2	21.1	27.7	38.2	13.6	21.6	26.2	15.1	28.2	40.7
	✓	20.3	23.9	29.5	22.5	30.9	43.2	13.9	22.3	28.0	15.8	30.8	43.9

simulation-to-real transfer of visual AI (especially semantic navigation).

The AI2THOR system provides a simulation framework, from which different synthesized inputs from the robot can be extracted and actions can be subsequently fed to the robot. In order to test OG-SGG, we collected 113 images in total from the 3 RoboTHOR validation scenes, in all 5 different configurations. Each image was captured using different material randomization, and the camera was moved so that at least 6 objects are visible within the view. Afterwards, ground truth scene graphs were automatically generated with fixed, hard-coded rules that leverage information about the objects provided directly by the system, such as absolute positions or container relationships. Another simple ontology was also created to ground the concepts present in these scene graphs.

We performed the same experiments carried out for the TERESA dataset, including the ablation study. Table 4 contains dataset statistics, which show a similar increase in dataset density after applying filtering, which brings it to a level close to that of the desired target dataset. On the other hand, the data augmentation process seems to not result in a significant increase in density. This is probably once again due to lack of higher level rules in the ontology that can deduce new triplets from existing ones. Table 5 likewise confirms the performance improvements brought by the filtered dataset, and the post-processing stage. The metrics corresponding to the model trained on augmented splits seem to be within margin of error compared to ontology-guided filtering without data augmentation – this is consistent with the aforementioned lack of increase in density. Interestingly, the choice of filtered indoor dataset here has clearly resulted in worse performance across the board, except in the baseline

comparison case. This might be due to the harsher filtering caused by the AI2THOR ontology, which has effectively halved the number of images available and thus might not be able to form a critical mass necessary for the network to be successfully trained.

Fig. 9 showcases a few qualitative examples, taking the VG-SGG split with filtering and no data augmentation as the basis for the trained model. It can be seen that in general the network has a tendency to be overly enthusiastic in predicting `has` and `near`, the former in the baseline and the latter in our improved system. In any case, it can be observed that ontology violations (such as (`Cup_1`,`has`,`Apple_1`) in the second image) do not appear in our improved output thanks to the post-processing.

VIII. ADDITIONAL EXPERIMENTS USING DIFFERENT MODELS

We decided to test OG-SGG with a different scene graph generation network, in order to further exemplify its model agnosticity. Specifically we prepared and tested two simple models (see Fig. 10) while reusing the existing training pipeline:

- **Simple Semantic model**: The two semantic vectors are concatenated and fed to a hidden fully connected layer of size 800, followed by ReLU activation, batch normalization and a final fully connected layer of size N (number of relationship predicates to be detected). The intention behind this model is focusing purely on language priors between object classes and relation classes; while ignoring all other inputs (i.e. object locations, image features, etc.).
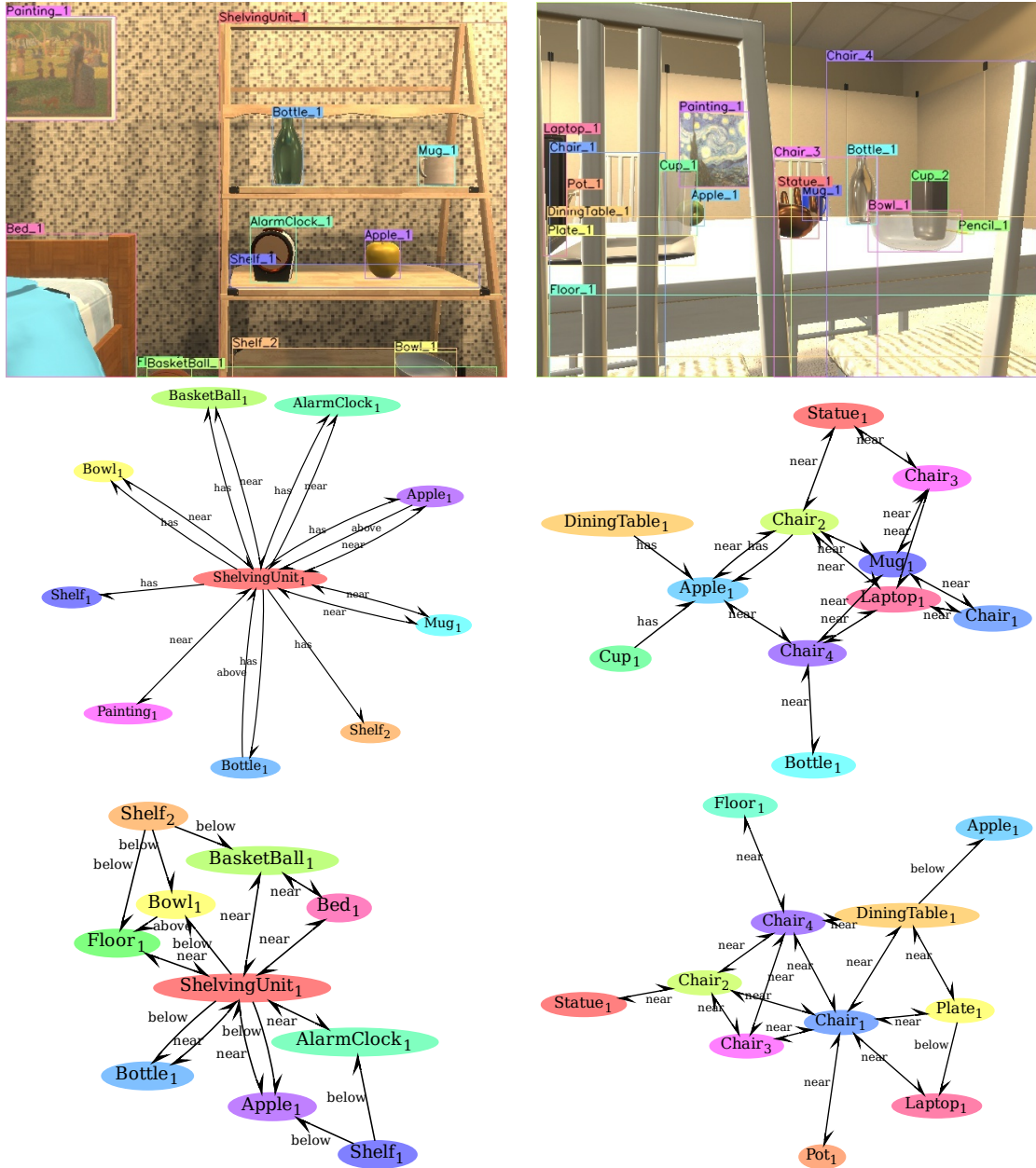
FIGURE 9. Qualitative results on AI2THOR test set. **Top row**: Image with object annotations. **Middle row**: "VG-SGG baseline." **Bottom row**: "VG-SGG with filter + post processing."

- **Simple Semantic-Positional model**: In addition to the above, an object mask processor component based on several CNN layers generates a new vector of the same size as the semantic vectors, which is concatenated alongside the two semantic vectors (forming the input of the hidden layer). The hidden layer is also expanded to 1200 neurons. The intention behind this model is augmenting the Simple Semantic model with object location information.

Table 6 shows the Simple Semantic model evaluated on TERESA, Table 7 shows the Simple Semantic model evaluated on AI2THOR, and Table 8 shows the Simple Semantic-

Positional model evaluated on AI2THOR. The first thing to note is the performance uplift delivered by the post-processing logic, further proving the usefulness of enforcing ontological axioms on generated scene graphs. In the TERESA dataset, the combined efforts of the filtering and the post-processing are able to easily outperform the baseline. Applying filtering without post-processing also generally produces improved results, however there are regressions in some **R@K** metrics with low **K** and **k** – we believe this to be a side effect of increasing generalization capability by reducing overfitting. In the case of AI2THOR, both simple models have trouble learning useful information through the

training process. As expressed in Section VII, this may be due to the harsher filtering caused by the AI2THOR ontology. Nevertheless, the Simple Semantic model is able to dominate in **mR@K**, indicating greater generalization capability. Interestingly, even though the Simple Semantic-Positional model performs better in its baseline without OG-SGG than the Simple Semantic model, this initial advantage is unable to materialize into improved across-the-board performance with full OG-SGG applied, only managing to improve when post-processing without a filtered dataset is used.

Another comparison was made between the OG-SGG enriched Simple Semantic model and the baseline performance of a non-trivial model (VRD-RANS). Despite having an order of magnitude fewer parameters (VRD-RANS: 25M parameters, Simple Semantic model: 1.3M parameters), the obtained performance clearly exceeds that of baseline VRD-RANS without OG-SGG. The same can be said for the Simple Semantic-Positional model, however the gap in parameter count (25M vs 9.2M) is smaller.

IX. CONCLUSION AND FUTURE WORK

In this work we joined the world of ontologies together with the world of scene graph generation, and showed how the strategies we proposed (using only filtering and processing based on the most common OWL axioms affecting predicates) can achieve quantitative and qualitative improvements in domain specific environments. Whereas existing scene graph generation networks (such as VRD-RANS) generate all possible pairs, the OG-SGG methodology 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 resulted in improved **mR@K**, especially when compared to the baseline (that is, the model trained on the original version of the dataset without OG-SGG). Evaluating the performance without graph constraint priors (i.e. by setting the graph constraint hyperparameter **k** to its highest allowed value) also produced better results. We also show how OG-SGG improves the results for different application scenarios (TERESA and AI2THOR datasets), and also for different scene graph generation models.

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 filtering. 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 (higher order) axioms 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 [17], or through incorporating neurosymbolic propositional and first order logic directly as part of the training process [4]. Simultaneous Localization and Mapping (SLAM) systems could be yet another area of interest for future work related to ontology-guided machine learning. Specifically, Semantic SLAM systems capable of segmenting rooms and labelling/tracking all objects within could be proposed, and this potentially involves the detection of relationships between objects in a similar way to the scene graph generation problem.

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

.

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, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58:82–115, 2020.

[4] Samy Badreddine, Artur d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks, 2022.

[5] Adrien Bennetot, Jean-Luc Laurent, Raja Chatila, and Natalia Díaz-Rodríguez. Towards explainable neural-symbolic visual reasoning.
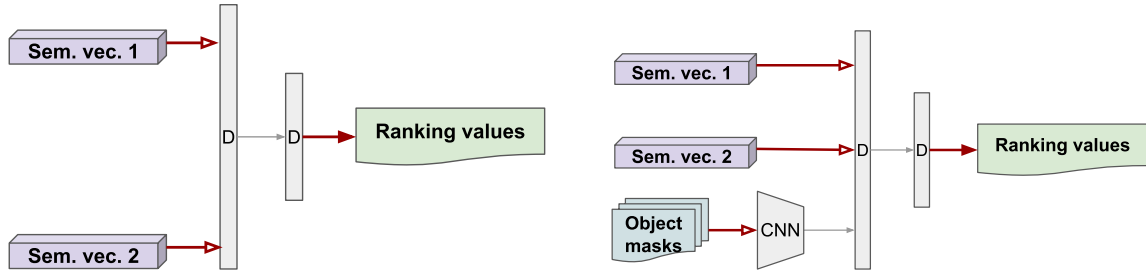
FIGURE 10. **Left: Simple Semantic model**, only containing two fully connected layers (marked with D). **Right: Simple Semantic-Positional model**, also containing a CNN module for processing the object position masks.

TABLE 6. TERESA evaluation results on **Trivial** model. The model was trained on 4 different dataset splits (2 for each source training dataset), and evaluated with and without post-processing (*"Post"* column). The baseline results for VRD-RANS trained on both source datasets are also shown in the table.

Dataset	Post	Metrics for Predicate Detection (PredDet)											
		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 unmodified dataset	✗	22.9	31.3	40.1	20.0	26.9	39.8	19.4	29.5	39.1	25.6	36.1	47.6
	✓	26.0	33.6	44.0	26.8	36.2	52.0	34.5	43.3	54.3	30.3	44.3	61.2
VG-SGG with filtering/augmentation	✗	21.5	25.0	35.8	19.4	31.2	42.8	31.3	40.2	45.6	31.9	42.5	47.9
	✓	**36.2**	**46.0**	**54.2**	**36.1**	**47.7**	**58.2**	**37.6**	**52.3**	**65.0**	**37.4**	**54.1**	**68.4**
VG-indoor unmodified dataset	✗	26.8	33.1	40.3	24.6	33.2	45.9	12.7	19.5	25.3	9.9	29.6	42.7
	✓	28.0	33.8	41.6	28.5	39.7	51.6	18.7	29.1	39.3	23.3	37.9	50.8
VG-indoor with filtering/augmentation	✗	**34.8**	36.3	42.1	29.4	39.7	46.6	27.9	36.4	52.3	24.8	37.9	52.2
	✓	34.5	**37.9**	**45.3**	**36.7**	**42.4**	**55.7**	**28.0**	**37.4**	**53.4**	**29.1**	**39.8**	**62.0**
VG-SGG unmodified (VRD-RANS)	✗	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-indoor unmodified (VRD-RANS)	✗	26.1	33.7	40.2	26.0	41.4	56.1	10.5	20.6	25.2	19.8	35.5	53.6

TABLE 7. AI2THOR evaluation results on **Trivial** model. The model was trained on 4 different dataset splits (2 for each source training dataset), and evaluated with and without post-processing (*"Post"* column). The baseline results for VRD-RANS trained on both source datasets are also shown in the table.

Dataset	Post	Metrics for Predicate Detection (PredDet)											
		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 unmodified dataset	✗	14.1	18.5	24.9	15.9	20.4	33.3	4.5	7.5	10.5	6.7	11.1	16.9
	✓	**17.3**	**25.6**	**36.3**	**18.0**	**26.1**	**42.9**	5.2	9.3	14.0	**7.7**	13.0	21.7
VG-SGG with filtering/augmentation	✗	4.8	10.3	14.7	6.8	14.9	24.9	5.8	9.6	13.6	6.0	12.0	20.3
	✓	5.0	11.4	18.0	8.3	17.7	32.4	**5.9**	**9.9**	**15.5**	7.0	**13.2**	**26.1**
VG-indoor unmodified dataset	✗	17.0	19.6	25.5	19.0	24.0	34.8	7.1	11.9	17.0	8.5	17.6	26.1
	✓	**19.4**	**25.0**	**32.9**	**21.2**	**28.7**	42.4	7.8	14.9	20.7	9.3	**20.1**	29.7
VG-indoor with filtering/augmentation	✗	16.6	21.7	27.8	18.0	23.8	36.2	9.5	15.0	19.7	10.2	18.6	28.7
	✓	17.3	24.6	32.7	19.0	28.1	**43.3**	**9.8**	**15.9**	**21.3**	10.4	20.0	**32.1**
VG-SGG unmodified (VRD-RANS)	✗	19.1	24.4	32.1	21.9	27.2	40.5	7.0	11.8	16.2	9.4	15.2	24.6
VG-indoor unmodified (VRD-RANS)	✗	18.7	21.4	26.0	22.1	26.5	37.4	9.3	17.7	23.0	12.2	22.3	31.5

IJCAI NeSy Workshop, arXiv preprint arXiv:1909.09065, 2019.

[6] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020.

[7] Guilhem Buisan, Guillaume Sarthou, 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.

[8] Ferenc Bálint-Benczédi, Jan-Hendrik Worch, Daniel Nyga, Nico Blodow, Patrick Mania, Z. Márton, and Michael Beetz. RoboSherlock: Cognition-enabled Robot Perception for Everyday Manipulation Tasks. ArXiv, abs/1911.10079, 2019.

[9] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Loddon Yuille. 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.

[10] Devleena Das, Siddhartha Banerjee, and Sonia Chernova. Explainable AI for Robot Failures: Generating Explanations that Improve User Assistance in Fault Recovery. In HRI '21: ACM/IEEE International Conference on Human-Robot Interaction, Boulder, CO, USA, March 8-11, 2021, pages 351–360, 03 2021.

[11] 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.

[12] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, Luca Weihs, Mark Yatskar, and Ali Farhadi. RoboTHOR: An Open Simulation-to-Real Embodied AI Platform. In CVPR, 2020.

[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] Ivan Donadello and Luciano Serafini. Integration of numeric and symbolic information for semantic image interpretation. Intelligenza Artificiale, 10(1):33–47, 2016.

[15] Ivan Donadello and Luciano Serafini. Compensating Supervision In-

IEEE Access

TABLE 8. AI2THOR evaluation results on **Trivial+** model. The model was trained on 4 different dataset splits (2 for each source training dataset), and evaluated with and without post-processing ("*Post*" column). The baseline results for VRD-RANS trained on both source datasets are also shown in the table.

Dataset	Post	Metrics for Predicate Detection (PredDet)											
		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 unmodified dataset	✗	12.3	18.3	26.1	16.7	25.5	40.8	12.9	17.1	20.7	15.6	23.6	32.1
	✓	**18.5**	**27.7**	**38.6**	**24.8**	**34.0**	**51.0**	**15.4**	19.9	24.3	**18.9**	27.5	37.8
VG-SGG with filtering/augmentation	✗	9.3	17.7	24.7	8.4	17.9	33.4	9.9	18.3	23.6	9.3	22.6	40.3
	✓	10.7	20.6	29.1	11.1	25.0	43.8	11.6	**20.4**	**26.0**	11.6	**27.8**	**48.2**
VG-indoor unmodified dataset	✗	21.5	24.0	27.1	23.5	28.6	38.5	10.6	18.4	21.7	11.7	22.9	31.6
	✓	**24.8**	**29.6**	**36.7**	**26.5**	**33.2**	**45.4**	**12.6**	20.7	25.4	13.0	25.5	36.6
VG-indoor with filtering/augmentation	✗	19.5	22.9	27.4	21.4	26.6	37.2	12.2	19.9	24.6	13.3	23.1	34.6
	✓	20.6	26.0	31.9	22.8	30.7	44.0	**12.6**	**21.4**	**27.0**	**13.9**	**26.2**	**39.9**
VG-SGG unmodified (VRD-RANS)	✗	19.1	24.4	32.1	21.9	27.2	40.5	7.0	11.8	16.2	9.4	15.2	24.6
VG-indoor unmodified (VRD-RANS)	✗	18.7	21.4	26.0	22.1	26.5	37.4	9.3	17.7	23.0	12.2	22.3	31.5

completeness with Prior Knowledge in Semantic Image Interpretation. In IJCNN, pages 1–8. IEEE, 2019.

[16] 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.

[17] Natalia Díaz-Rodríguez, Alberto Lamas, Jules Sanchez, Gianni Franchi, Ivan Donadello, Siham Tabik, David Filliat, Policarpo 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.

[18] Nicola Guarino, Daniel Oberle, and Steffen Staab. What Is an Ontology?, pages 1–17. Springer, 05 2009.

[19] 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.

[20] IEEE. IEEE Standard Ontologies for Robotics and Automation. IEEE Std 1872-2015, pages 1–60, 2015.

[21] 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.

[22] Petteri Karvinen, Natalia Díaz-Rodríguez, Stefan Grönroos, and Johan Lilius. RDF stores for enhanced living environments: an overview. Enhanced Living Environments, pages 19–52, 2019.

[23] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. arXiv, 2017.

[24] Rajat Koner, Poulami Sinhamahapatra, and Volker Tresp. Relation transformer network. arXiv preprint arXiv:2004.06193, 2020.

[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(1):32–73, 2017.

[26] Annica Kristoffersson, Silvia Coradeschi, and Amy Loutfi. A Review of Mobile Robotic Telepresence. Adv. in Hum.-Comp. Int., 2013:3:3–3:3, January 2013.

[27] Jean-Baptiste Lamy. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. Artificial Intelligence in Medicine, 80:11–28, 2017.

[28] Kongming Liang, Yuhong Guo, Hong Chang, and Xilin Chen. Visual Relationship Detection with Deep Structural Ranking. In AAAI Conference on Artificial Intelligence, 2018.

[29] Y. Liang, Y. Bai, W. Zhang, X. Qian, L. Zhu, and T. Mei. VrR-VG: Refocusing Visually-Relevant Relationships. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 10402–10411, Los Alamitos, CA, USA, nov 2019. IEEE Computer Society.

[30] 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 Pajdla,

Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision – ECCV 2014, pages 740–755, Cham, 2014. Springer International Publishing.

[31] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander Berg. SSD: Single Shot MultiBox Detector. In Computer Vision – ECCV 2016, volume 9905, pages 21–37, 10 2016.

[32] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual Relationship Detection with Language Priors. In European Conference on Computer Vision, 2016.

[33] George A. Miller. WordNet: A Lexical Database for English. Commun. ACM, 38(11):39–41, November 1995.

[34] Mark A. Musen. The Protégé project: A look back and a look forward. AI Matters, 1(4):4–12, 2015.

[35] Alberto Olivares-Alarcos, Daniel Beßler, Alaa Khamis, Paulo Goncalves, Maki K Habib, Julita Bermejo-Alonso, Marcos Barreto, Mohammed Diab, Jan Rosell, João Quintas, et al. A review and comparison of ontology-based approaches to robot autonomy. The Knowledge Engineering Review, 34, 2019.

[36] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. ROS: an open-source Robot Operating System. In ICRA Workshop on Open Source Software, 2009.

[37] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779–788, 2016.

[38] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 28. Curran Associates, Inc., 2015.

[39] Guillaume Sarthou, Aurélie Clodic, and Rachid Alami. Ontologenius: A long-term semantic memory for robotic agents. In 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), pages 1–8, 2019.

[40] Arne Seeliger, Matthias Pfaff, and Helmut Krcmar. Semantic Web Technologies for Explainable Machine Learning Models: A Literature Review. PROFILES/SEMEX@ ISWC, 2465:1–16, 2019.

[41] Sahand Sharifzadeh, Sina Moayed Baharlou, and Volker Tresp. Classification by Attention: Scene Graph Classification with Prior Knowledge. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, 2020.

[42] K. Shiarlis, J. Messias, M. van Someren, S. Whiteson, J Kim, J Vroon, G. Englebienne, K. Truong, V. Evers, N. Perez-Higueras, I. Perez-Hurtado, R. Ramon-Vigo, F. Caballero, L. Merino, J. Shen, S. Petridis, M. Pantic, L. Hedman, M. Scherlund, R. Koster, and H. Michel. TERESA: A Socially Intelligent Semi-autonomous Telepresence System. In Workshop on Machine Learning for Social Robotics at ICRA-2015 in Seattle, 2015.

[43] K. Tang, Y. Niu, J. Huang, J. Shi, and H. Zhang. Unbiased Scene Graph Generation From Biased Training. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 3713–3722, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society.

[44] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei

IEEE *Access*

Liu. Learning to Compose Dynamic Tree Structures for Visual Contexts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[45] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The New Data in Multimedia Research. Commun. ACM, 59(2):64–73, January 2016.

[46] D. Tsarkov and I. Horrocks. FaCT++ Description Logic Reasoner: System Description. In Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006), volume 4130 of Lecture Notes in Artificial Intelligence, pages 292–297. Springer, 2006.

[47] Katherine M Tsui, Munjal Desai, Holly A Yanco, and Chris Uhlik. Exploring use cases for telepresence robots. In 2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 11–18. IEEE, 2011.

[48] W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview (Second Edition) - W3C Recommendation 11 December 2012. Technical report, W3C, December 2012.

[49] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 1571–1580, 2020.

[50] Lei Wang, Peizhen Lin, Jun Cheng, Feng Liu, Xiaoliang Ma, and Jianqin Yin. Visual relationship detection with recurrent attention and negative sampling. Neurocomputing, 434:55–66, 2021.

[51] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. Scene Graph Generation by Iterative Message Passing. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3097–3106, Los Alamitos, CA, USA, jul 2017. IEEE Computer Society.

[52] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, Attend and 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.

[53] Ruichi Yu, Ang Li, Vlad Morariu, and Larry Davis. Visual Relationship Detection with Internal and External Linguistic Knowledge Distillation. In IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pages 1068–1076, 10 2017.

[54] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural Motifs: Scene Graph Parsing with Global Context. In Conference on Computer Vision and Pattern Recognition, 2018.
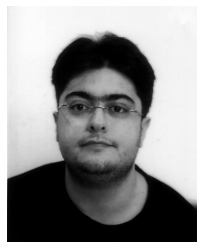
FERNANDO CABALLERO is PhD on Robotics in 2007 from the University of Seville (Spain). He is currently Associate Professor at the University of Seville, and co-leader of the Service Robotics Laboratory. His research work is focused on localization, perception, and planning for mobile robots, both ground and aerial. He is co-author of more than one hundred publications in international conferences and journals.

NATALIA DÍAZ-RODRÍGUEZ is double PhD (2015) from University of Granada (Spain) and Abo Akademi University (Finland). She is currently researcher and docent at the DaSCI Andalusian Research Institute in data science and computational intelligence at the Dept. of Computer Science and Artificial Intelligence. Earlier, she worked in Silicon Valley, CERN, Philips Research, University of California Santa Cruz and with FDL Programme with NASA. She was also Assistant Prof. of Artificial Intelligence at the Autonomous Systems and Robotics Lab (U2IS) at ENSTA, Institut Polytechnique Paris, INRIA Flowers team on developmental robotics during 4 years, and worked on open-ended learning and continual/lifelong learning for applications in computer vision and robotics. Her current research interests include deep learning, explainable Artificial Intelligence (XAI), Responsible AI and AI for social good. Her background is on knowledge engineering and is interested in neural-symbolic approaches to practical applications of AI.

FERNANDO AMODEO is currently a PhD student at Pablo de Olavide University (Spain). He received a Bachelor's Degree in Software Engineering (2016) and a Master's Degree in Logic, Computation and Artificial Intelligence (2022) from the University of Seville (Spain); earning Honours in both of his final projects.

He is currently a researcher at the Service Robotics Lab focused on artificial intelligence and computer vision. His current research interests include deep learning and neuro-symbolic approaches applied to robotics.

LUIS MERINO (M'15). MSc Degree on Telecommunications Engineering (2000), PhD in Robotics (2007), from the University of Seville, Spain. His PhD won the ABB Award to the Best Doctoral Dissertation on Robotics in Spain, given by the Spanish Committee of Automation. He is currently Associate Professor at Universidad Pablo de Olavide (UPO), Seville, Spain, where he co-leads the Service Robotics Laboratory. He has been Vice-Dean of the School of Engineering for 5 years. His interest includes robot autonomous navigation, including human-aware navigation, decision-making and control for social robotics, planning under uncertainties and multi-robot systems. He has published more than 90 papers and has led UPO's team in several national and international projects on those topics. Prof. Merino is member of the IEEE, the Spanish Committee on Automation, and of the International Socially Intelligent Robotics Consortium. He is Associate Editor of major robotics conferences like IEEE ICRA and IROS conferences, and of the Image and Vision Computing journal.

• • •