

Improving Scene Graph Classification by Exploiting Knowledge from Texts

Sahand Sharifzadeh,^{1*} Sina Moayed Baharlou,^{1*†} Martin Schmitt,²
Hinrich Schütze,² Volker Tresp^{1,3}

¹ Department of Informatics, LMU Munich, Germany

² Center for Information and Language Processing (CIS), LMU Munich, Germany

³ Siemens AG, Munich, Germany

sahand.sharifzadeh@gmail.com, sina.baharlou@gmail.com

Abstract

Training scene graph classification models requires a large amount of annotated image data. Meanwhile, scene graphs represent relational knowledge that can be modeled with symbolic data from texts or knowledge graphs. While image annotation demands extensive labor, collecting textual descriptions of natural scenes requires less effort. In this work, we investigate whether textual scene descriptions can substitute for annotated image data. To this end, we employ a scene graph classification framework that is trained not only from annotated images but also from symbolic data. In our architecture, the symbolic entities are first mapped to their correspondent image-grounded representations and then fed into the relational reasoning pipeline. Even though a structured form of knowledge, such as the form in knowledge graphs, is not always available, we can generate it from unstructured texts using a transformer-based language model. We show that by fine-tuning the classification pipeline with the extracted knowledge from texts, we can achieve $\sim 8x$ more accurate results in scene graph classification, $\sim 3x$ in object classification, and $\sim 1.5x$ in predicate classification, compared to the supervised baselines with only 1% of the annotated images.

Introduction

Relational reasoning is one of the essential components of intelligence; humans explore their environment by grasping the entire context of a scene rather than studying each item in isolation from the others. Furthermore, we expand our understanding of the world by educating ourselves about novel facts through reading or listening. For example, we might have never seen a “cow wearing a dress” but might have read about Hindu traditions of decorating cows. While we already have a robust visual system that can extract basic visual features such as edges and curves from a scene, the description of a “cow wearing a dress” refines our visual understanding of relations on an object level and enables us to recognize a dressed cow when seeing it.

Relational reasoning is gaining growing popularity in the Computer Vision community and especially in the form of

*These authors contributed equally.

†S. M. Baharlou contributed to this project while he was a visiting researcher at the Ludwig Maximilian University of Munich. Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

scene graph (SG) classification. The goal of SG classification is to classify objects and their relations in an image. One of the challenges in SG classification is collecting annotated image data. Most approaches in this domain rely on thousands of manually labeled and curated images. In this paper, we investigate whether the SG classification models can be fine-tuned from textual scene descriptions (similar to the “dressed cow” example above).

We consider a classification pipeline with two major parts: a feature extraction *backbone*, and a *relational reasoning* component (Figure 1). The backbone is typically a convolutional neural network (CNN) that detects objects and extracts an image-based representation for each. On the other hand, the relational reasoning component can be a variant of a recurrent neural network [Xu et al. 2017, Zellers et al. 2018] or graph convolutional networks [Yang et al. 2018, Sharifzadeh, Baharlou, and Tresp 2021]. This component operates on an object level by taking the latent representations of all the objects in the image and propagating them in the graph.

Note that, unlike the feature extraction backbone that requires images as input, the relational reasoning component operates on graphs with the nodes representing objects and the edges representing relations. The distinction between the input to the backbone (images) and the relational reasoning component (graphs) is often overlooked. Instead, the scene graph classification pipeline is treated as a network that takes only images as inputs. However, one can also train or fine-tune the relational reasoning component directly by injecting it with relational knowledge. For example, Knowledge Graphs (KGs) contain curated facts that indicate the relations between a `head` object and a `tail` object in the form of (`head`, `predicate`, `tail`) e.g., (`Person`, `Rides`, `Horse`). The facts in KGs are represented by symbols whereas the inputs to the relational reasoning component are image-based embeddings. In this work, we map the triples to image-grounded embeddings as if they are coming from an image. We then use these embeddings to fine-tune the relational reasoning component through a denoising graph autoencoder scheme.

Note that the factual knowledge is not always available in a well-structured form, specially in domains where the knowledge is not stored in the machine-accessible form of KGs. In fact, most of the collective human knowledge is only

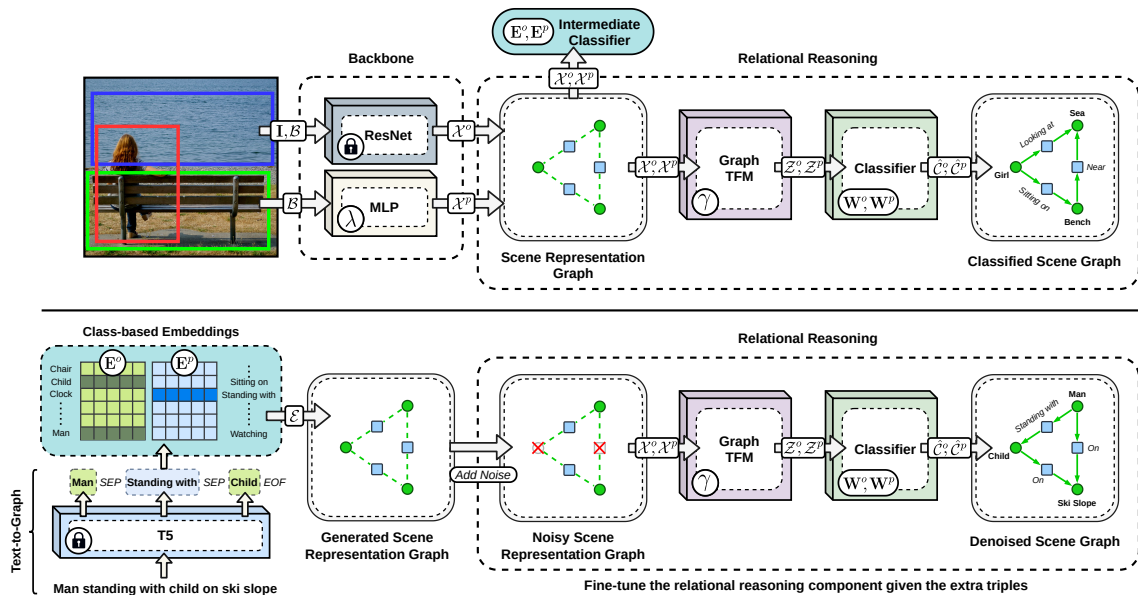


Figure 1: Top: we initially train a scene graph classification pipeline from images and their corresponding SGs. Bottom: we then use a text-to-graph module to extract structured knowledge from unstructured texts. The extracted graph is embedded by image-grounded vectors, masked, and then fed to the relational reasoning module to predict the missing relations and thus, encourage the network to learn the new relations from texts. The *lock* sign indicates pre-trained and frozen parts of the network.

available in the unstructured form of texts and documents. Exploiting this form of knowledge, in addition to structured knowledge, can be significantly beneficial. To this end, we employ a transformer-based model to generate structured graphs from textual input and utilize them to improve the relational reasoning module.

In summary, we propose *Texema*, a scene graph classification pipeline that can be trained from the large corpora of unstructured knowledge. We evaluate our approach on the Visual Genome dataset. In particular, we show that we can fine-tune the reasoning component using textual scene descriptions instead of thousands of images. As a result, when using as little as ~ 500 images (1% of the VG training data), we can achieve $\sim 3x$ more accurate results in object classification, $\sim 8x$ in scene graph classification and $\sim 1.5x$ in predicate classification compared to the supervised baselines. Additionally, in our ablation studies, we evaluate the performance of using different rule-based, LSTM-based, and transformed-based text-to-graph models.

Related Works

Scene Graph Classification: There is an extensive body of work on visual reasoning in general that includes different forms of reasoning [Wu, Lenz, and Saxena 2014, Deng et al. 2014, Hu et al. 2016, 2017, Santoro et al. 2017, Zellers et al. 2019]. Here, we mainly review the works that are focused on scene graph classification. Visual Relation Detection (VRD) [Lu et al. 2016] and the Visual Genome [Krishna et al. 2017] are the main datasets for this task. While the original papers on VRD and VG provide the baselines for scene graph classification by treating objects indepen-

dently, several follow-up works contextualize the entities before classification. Iterative Message Passing (IMP) [Xu et al. 2017], Neural Motifs [Zellers et al. 2018] (NM), Graph R-CNN [Yang et al. 2018], and Schemata [Sharifzadeh, Baharlou, and Tresp 2021] proposed to propagate the image context using basic RNNs, LSTMs, graph convolutions, and graph transformers respectively. On the other hand, authors of VTransE [Zhang et al. 2017] proposed to capture relations by applying TransE [Bordes et al. 2013], a knowledge graph embedding model, on the visual embeddings, Tang et al. [2019] exploited dynamic tree structures to place the object in an image into a visual context. Chen et al. [2019a] proposed a multi-agent policy gradient method that frames objects into cooperative agents and then directly maximizes a graph-level metric as the reward. In tangent to those works, Sharifzadeh et al. [2021] proposed to enrich the input domain in scene graph classification by employing the predicted pseudo depth maps of VG images that were released as an extension called *VG-Depth*.

Commonsense in Scene Understanding: Several recent works have proposed to employ external or internal sources of knowledge to improve visual understanding [Wang, Ye, and Gupta 2018, Jiang et al. 2018, Singh et al. 2018, Kato, Li, and Gupta 2018]. In the scene graph classification domain, some of the works have proposed to correct the SG prediction errors by merely comparing them to the co-occurrence statistics of internal triples as a form of commonsense knowledge [Chen et al. 2019c,b, Zellers et al. 2018]. Earlier, Baier, Ma, and Tresp [2017, 2018] proposed the first scene graph classification model that employed prior knowledge in the form of Knowledge Graph Embeddings (KGEs) that gen-

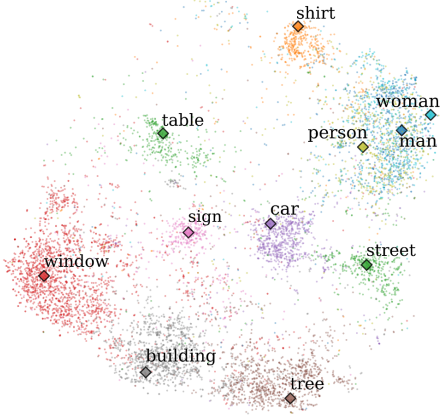


Figure 2: The t-SNE representation of the e_i s (diamonds) and image-based representations \mathcal{X} s (dots) where each color represents the ground-truth class of the dot.

eralize beyond the given co-occurrence statistics. Zareian, Karaman, and Chang [2020], Zareian et al. [2020] followed this approach by extending it to models that are based on graph convolutional networks. More recently, Sharifzadeh, Baharlou, and Tresp [2021] proposed Schemata as a generalized form of a KGE model that is learned directly from the images rather than triples. In general, scene graph classification methods are closely related to the KGE models. Therefore, we refer the interested readers to [Nickel et al. 2016, Ali et al. 2020a,b] for a review and large-scale study on the KG models, and to [Tresp, Sharifzadeh, and Konopatzki 2019, Tresp et al. 2020] for an extensive investigation of the connection between perception, KG models, and cognition.

Nevertheless, to the best of our knowledge, the described methods have employed curated knowledge in the form of triples, and none of them have directly exploited the textual knowledge. In this direction, the closest work to ours is by Yu et al. [2017], proposing to distill the external language knowledge using a teacher-student model. However, this work does not include a relational reasoning component and only refines the final predictions. Also, as shown in the experiments, our knowledge extraction module performs two times better than the SG Parser used in that work.

Knowledge Extraction from Text: Knowledge extraction from text has been studied for a long time [Chinchor 1991]. Previous work ranges from pattern-based approaches [Hearst 1992] to supervised neural approaches with specialized architectures [Gupta et al. 2019, Yaghoobzadeh, Adel, and Schütze 2017]. Recently, Schmitt et al. [2020] successfully applied a general sequence-to-sequence architecture to graph \leftrightarrow text conversion. With the recent rise of transfer learning in NLP, an increasing number of approaches are based on large language models, pre-trained in a self-supervised manner on massive amounts of texts [Devlin et al. 2019]. Inspired from previous work that explores transfer learning for graph-to-text conversion [Ribeiro et al. 2020], we base our text-to-graph model on a pre-trained T5 model [Raffel et al. 2019].

Input	man standing with child on ski slope
Reference	(child, on, ski slope) (man, on, ski slope)
Graph (RG)	(man, standing with, child)
R _{text\rightarrowgraph}	(man, standing, child)
SSGP	(standing, with, child) (standing, on, slope)
CopyNet (1%)	(man, standing with, child)
T5 (1%)	(man, standing with, child)
CopyNet (10%)	(man, standing with, child) (child, on, slope)
T5 (10%)	(man, standing with, child) (child, on, ski slope)

Table 1: An example of extracted triples from a given text input in VG, using different methods. Green: correct (\in RG). Red: incorrect (\notin RG). The results are computed using the respective official code bases of the related works.

Methods

In this section, we first describe the backbone and relational reasoning components. We then describe our approach for fine-tuning the network from texts. We have three possible forms of data: Images (**IM**), Scene Graphs (**SG**) and Textual Scene Descriptions (**TXT**). We consider having two sets of data: one is the *parallel* set, which is the set of IM with their corresponding SG and TXT, and another is the *text* set which is a set of additional TXT that come without any images or scene graphs. These two sets have no elements in common.

We initially train our backbone and relational reasoning component from IM and SG, and our text-to-graph model from the TXT and SG in the parallel set. We then show that we can fine-tune the pipeline using the text set and without using any additional images.

Backbone (Algorithm 1.1)

The feature-extraction backbone is a convolutional neural network (ResNet-50) that has been pre-trained in a self-supervised manner [Grill et al. 2020] from unlabeled images of ImageNet [Deng et al. 2009] and Visual Genome [Krishna et al. 2017]. Given an image **I** with several objects in bounding boxes $\mathcal{B} = \{\mathbf{b}_i\}_{i=1}^n$, $\mathbf{b}_i = [b_i^x, b_i^y, b_i^w, b_i^h]$, we apply the ResNet-50 to extract pooled object features $\mathcal{X}^o = \{\mathbf{x}_i^o\}_{i=1}^n$, $\mathbf{x}_i^o \in \mathbb{R}^d$. Here $[b_i^x, b_i^y]$ are the coordinates of \mathbf{b}_i and $[b_i^w, b_i^h]$ are its width and height, and d are the vector dimensions. Following [Zellers et al. 2018], we define $\mathcal{X}^p = \{\mathbf{x}_i^p\}_{i=1}^m$, $\mathbf{x}_i^p \in \mathbb{R}^d$ as the relational features between each pair of objects. Each \mathbf{x}_i^p is initialized by applying a two layered fully connected network on the relational position vector \mathbf{t} between a head i and a tail j where $\mathbf{t} = [t_x, t_y, t_w, t_h]$, $t_x = (b_i^x - b_j^x)/b_i^w$, $t_y = (b_i^y - b_j^y)/b_j^h$, $t_w = \log(b_i^w/b_j^w)$, $t_h = \log(b_i^h/b_j^h)$. The implementation and pre-training details of the layers are provided in the Evaluation. \mathcal{X}^o and \mathcal{X}^p form a structured presentation of the objects and predicates in the image also known as

Algorithm 1: Classify objects/predicates from images

1. Extract image features (Backbone):**Input:** Images and object bounding boxes $(\mathbf{I}, \mathcal{B} : \{\mathbf{b}_i\}_{i=1}^n)$.**Output:** Object embeddings $\mathcal{X}^o : \{\mathbf{x}_i^o\}_{i=1}^n$ and predicate embeddings $\mathcal{X}^p : \{\mathbf{x}_i^p\}_{i=1}^m$.**Trainable params:** λ .

$$\mathcal{X}^o = ResNet50(\mathbf{I}, \mathcal{B})$$

$$\mathcal{X}^p = \{MLP_\lambda(t(\mathbf{b}_i, \mathbf{b}_j)) \mid \forall \mathbf{b}_i, \mathbf{b}_j \in \mathcal{B}\}$$

2. Contextualize and Classify (Relational Reasoning):**Input:** Object embeddings $\mathcal{X}^o : \{\mathbf{x}_i^o\}_{i=1}^n$, Predicate embeddings $\mathcal{X}^p : \{\mathbf{x}_i^p\}_{i=1}^m$ and ground truth classes \mathcal{C}^o and \mathcal{C}^p .**Output:** Predicted object class distribution $\hat{\mathcal{C}}^o : \{\hat{\mathbf{c}}_i^o\}_{i=1}^n$ and predicted predicate class distribution $\hat{\mathcal{C}}^p : \{\hat{\mathbf{c}}_i^p\}_{i=1}^m$.**Trainable params:** $\gamma, \mathbf{W}^o, \mathbf{W}^p$.

$$\mathcal{Z}^o, \mathcal{Z}^p = GraphTransformer_\gamma(\mathcal{X}^o, \mathcal{X}^p)$$

$$\hat{\mathcal{C}}^o = \{\text{softmax}(\mathbf{W}^o \cdot \mathbf{z}^o) \mid \forall \mathbf{z}^o \in \mathcal{Z}^o\}$$

$$\hat{\mathcal{C}}^p = \{\text{softmax}(\mathbf{W}^p \cdot \mathbf{z}^p) \mid \forall \mathbf{z}^p \in \mathcal{Z}^p\}$$

3. Apply Loss (Cross-Entropy):

$$l_o = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \|\mathbf{c}_{i,j}^o\| \mathbf{c}_{i,j}^o \cdot \log(\hat{\mathbf{c}}_{i,j}^o)$$

$$l_p = -\frac{1}{m} \sum_{i=1}^m \sum_{j=1}^m \|\mathbf{c}_{i,j}^p\| \mathbf{c}_{i,j}^p \cdot \log(\hat{\mathbf{c}}_{i,j}^p)$$

Scene Representation Graph (SRG) [Sharifzadeh, Baharou, and Tresp 2021]. SRG is a fully connected graph with each node representing either an object or a predicate, where each object node is a direct neighbor to predicate nodes and each predicate node is a direct neighbor with its head and tail object nodes.

Relational Reasoning (Algorithm 1.2)

The relational reasoning component updates the initial SRG representations through Graph Transformer layers [Koncel-Kedziorski et al. 2019]. The outputs of these layers are $\mathcal{Z}^o = \{\mathbf{z}_i^o\}_{i=1}^n, \mathbf{z}_i^o \in \mathbb{R}^d$ and $\mathcal{Z}^p = \{\mathbf{z}_i^p\}_{i=1}^m, \mathbf{z}_i^p \in \mathbb{R}^d$ with equal dimensions as \mathcal{X} s. From here on, we drop the superscripts of o and p for brevity. We apply a linear classification layer \mathbf{W} to classify the contextualized representations \mathcal{Z} such that $\hat{\mathbf{c}} = \text{softmax}(\mathbf{W} \cdot \mathbf{z}_i)$, with cross-entropy as the loss function.

Fine-tuning from Texts (Algorithm 2)

Let us assume that we have already trained the backbone and relational reasoning components from IM and SG in the *parallel* set. Now, we want to fine-tune the weights in the relational reasoning component given the additional *text* set. The relational reasoning component takes graphs as input, therefore, we first need to convert TXT to SG:

Text-to-graph: This model is trained from the SG and TXT in the parallel set, and then used to generate SG from the text set. Let us consider an unstructured text such as “man standing with child on ski slope” (Table 1 - Input). A structured form of this sentence is a graph with unique nodes and edges for each entity or

Algorithm 2: Fine-tune the relational reasoning component from textual triples using a denoising auto-encoder paradigm

1. Learn image-grounded representations \mathbf{E} for each symbol through classification (without Graph Transformer):**Input:** Object embeddings $\mathcal{X}^o : \{\mathbf{x}_i^o\}_{i=1}^n$, predicate embeddings $\mathcal{X}^p : \{\mathbf{x}_i^p\}_{i=1}^m$ and their corresponding ground truth classes \mathcal{C}^o and \mathcal{C}^p .**Output:** Predicted object class distribution $\hat{\mathcal{C}}^o : \{\hat{\mathbf{c}}_i^o\}_{i=1}^n$ and predicted predicate class distribution $\hat{\mathcal{C}}^p : \{\hat{\mathbf{c}}_i^p\}_{i=1}^m$.**Trainable params:** $\mathbf{E}^o, \mathbf{E}^p$.

$$\hat{\mathcal{C}}^o = \{\text{softmax}(\mathbf{E}^o \cdot \mathbf{x}^o) \mid \forall \mathbf{x}^o \in \mathcal{X}^o\}$$

$$\hat{\mathcal{C}}^p = \{\text{softmax}(\mathbf{E}^p \cdot \mathbf{x}^p) \mid \forall \mathbf{x}^p \in \mathcal{X}^p\}$$

2. Apply Loss (Cross Entropy):

$$l_o = -\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^n \|\mathbf{c}_{i,j}^o\| \mathbf{c}_{i,j}^o \cdot \log(\hat{\mathbf{c}}_{i,j}^o)$$

$$l_p = -\frac{1}{m} \sum_{i=1}^m \sum_{j=1}^m \|\mathbf{c}_{i,j}^p\| \mathbf{c}_{i,j}^p \cdot \log(\hat{\mathbf{c}}_{i,j}^p)$$

3. Fine-tune the relational reasoning component given the extra triples (Denoising Graph Autoencoder):**Input:** Symbolic triples $\mathcal{S} : \{(h_i, p_i, t_i)\}_{i=1}^k$ and canonical object/predicate representations $\mathbf{E}^o/\mathbf{E}^p$.**Output:** Embedded representations $\mathcal{E} : \{(\mathbf{e}_i^h, \mathbf{e}_i^p, \mathbf{e}_i^t)\}_{i=1}^k$.**Trainable params:** $\gamma, \mathbf{W}^o, \mathbf{W}^p$.

- Build $\mathcal{E} : \{(\mathbf{e}_i^h, \mathbf{e}_i^p, \mathbf{e}_i^t)\}_{i=1}^k$ where for each (h_i, p_i, t_i) :
 $\mathbf{e}_i^h = \text{onehot}(h_i) \cdot \mathbf{E}^o$
 $\mathbf{e}_i^p = \text{onehot}(p_i) \cdot \mathbf{E}^p$
 $\mathbf{e}_i^t = \text{onehot}(t_i) \cdot \mathbf{E}^o$
 - Randomly set 20% of the nodes and edges in \mathcal{E} to zero.
 - Set $\mathcal{X}^o = \mathcal{E}^h \cup \mathcal{E}^t$ and $\mathcal{X}^p = \mathcal{E}^p$ and run Algorithm 1.2 to fine-tune $\gamma, \mathbf{W}^o, \mathbf{W}^p$, with $\mathcal{E}^h, \mathcal{E}^t$ and \mathcal{E}^p as the set of all heads, tails, and predicates in \mathcal{E} .
-

predicate. For example, the reference graph for this sentence contains the triples (child, on, ski slope), (man, standing with, child) and (man, on, ski slope) (Table 1 - RG).

In order to learn this mapping, we employ a transformer-based [Vaswani et al. 2017] sequence-to-sequence T5_{small} model [Raffel et al. 2019] and adapt it for the task of extracting graphs from texts. T5 consists of an encoder with several layers of self-attention (like BERT, Devlin et al. 2019) and a decoder with autoregressive self-attention (like GPT-3, Brown et al. 2020). In order to use a T5 model with graphs, we need to represent the graphs as a sequence. To this end, we serialize the graphs by writing out their facts separated by end-of-fact symbols (EOF), and separate the elements of each fact with SEP symbols [Schmitt et al. 2020], e.g. “child SEP on SEP ski slope EOF” (Fig. 1). To adapt the multi-task setting from T5’s pretraining, we use the task prefix “make graph: ” to mark our text-to-graph task. Table 1 shows an example text and the extracted graphs using T5 and other previous methods (see Evaluation for details).

Map to embeddings: Note that the predicted graphs are a sequence of symbols for heads, predicates, and tails where

Method	Precision		Recall		F1	
	1%	10%	1%	10%	1%	10%
R _{text→graph}	1.92 ± 0.00	1.86 ± 0.01	1.87 ± 0.00	1.81 ± 0.01	1.89 ± 0.00	1.84 ± 0.01
SSGP	14.86 ± 0.01	14.52 ± 0.02	18.47 ± 0.01	18.05 ± 0.02	16.47 ± 0.01	16.09 ± 0.02
CopyNet	29.20 ± 0.13	30.77 ± 0.49	27.19 ± 0.28	29.79 ± 0.29	28.16 ± 0.21	30.27 ± 0.34
T5	33.37 ± 0.11	33.81 ± 0.08	31.06 ± 0.18	32.45 ± 0.33	32.17 ± 0.13	33.12 ± 0.16

Table 2: The mean and standard deviation of Precision, Recall, and F1 scores of the predicted facts from the texts on four random splits. The results are computed using the respective official code bases of the related works and evaluated on VG.

each symbol represents a class $c \in \mathcal{C}$. However, the inputs to the relational reasoning component are image-based vectors \mathcal{X} . Thus, before feeding the symbols to the relational reasoning component, we need to map them to a corresponding embedding from the space of \mathcal{X} as if we are feeding it with image-based embeddings. In order to do that, we train a mapping from symbols to \mathcal{X} s using the IM and SG of the parallel set. This is simply done by training a linear classification layer \mathbf{E} given \mathcal{X} s from the parallel set (Algorithm 2.1). Unlike the classification layer in Algorithm 1, here we classify \mathcal{X} s instead of \mathcal{Z} s and the goal is *not* to use the classification output but to train image-grounded, canonical representations for each class: each row e_i in the classification layer becomes a cluster center for \mathcal{X} s from class i (Figure 2). Therefore, instead of the extracted symbolic c_i from the text set, we can feed its canonical image-grounded representation e_i to the graph transformer (Algorithm 2.3).

Denoising Graph Autoencoder: To fine-tune the relational reasoning given this data, we treat the relational reasoning component as a denoising autoencoder where the input is an incomplete (noisy) graph that comes from the text and the output is the denoised graph. If we do not apply a denoising autoencoder paradigm, the function will collapse to an identity map. We create the noisy graph by randomly setting some of the input nodes and edges to zero during the training (Algorithm 2.3). The goal is to encourage the graph transformer to predict the missing links and therefore, learn the relational structure.

Evaluation

We first compare the performance of different rule-based and embedding-based text-to-graphs models on our data. We then evaluate the performance of our entire pipeline in classifying objects and relations in images. In particular, we show that the extracted knowledge from the texts can largely substitute annotated images as well as ground-truth graphs.

Dataset We use the sanitized version [Xu et al. 2017] of Visual Genome (VG) dataset [Krishna et al. 2017] including images and their annotations, i.e., bounding boxes, scene graphs, and scene descriptions. Our goal is to design an experiment that evaluates whether we can substitute annotated images with textual scene descriptions. Therefore, instead of using external textual datasets with unbounded information, we use Visual Genome itself by dividing it into different splits of *parallel* (with IM, SG and TXT) and *text* data (with only

TXT). To this end, we assume only a random proportion (1% or 10%) of training images are annotated (parallel set containing IM with corresponding SG and TXT). We consider the remaining data (99% or 90%) as our text set and discard their IM and SG. We aim to see whether employing TXT from the text set, can substitute the discarded IM and SG from this set. We use four different random splits [Sharifzadeh, Baharlou, and Tresp 2021] to avoid a sampling bias. For more detail on the datasets refer to the supplementary materials.

Note that the scene graphs and the scene descriptions from the VG are collected separately and by crowd-sourcing. Therefore, even though the graphs and the scene descriptions refer to the same image region, they are disjoint and contain complementary knowledge.

Graphs from Texts

The goal of this experiment is to study the effectiveness of the text-to-graph model. We fine-tune the pre-trained T5 model on parallel TXT and SG, and apply it on the text set to predict their corresponding SG. We also implement the following rule-based and embedding-based baselines to compare their performance using our splits: (1) R_{text→graph} is a simple rule-based system introduced by Schmitt et al. [2020] for general knowledge graph generation from text. (2) The Stanford Scene Graph Parser (SSGP) [Schuster et al. 2015] is another rule-based approach that is more adapted to the scene graph domain. Even though this approach was not specifically designed to match the scene graphs from the Visual Genome dataset, it was still engineered to cover typical idiosyncrasies of textual image descriptions and corresponding scene graphs. (3) CopyNet [Gu et al. 2016] is an LSTM sequence-to-sequence model with a dedicated copy mechanism, which allows copying text elements directly into the graph output sequence. It was used for unsupervised text-to-graph generation by Schmitt et al. [2020]. However, we train it on the supervised data of our parallel sets. We use a vocabulary of around 70k tokens extracted from the VG-graph-text benchmark [Schmitt et al. 2020] and, otherwise, also adopt the hyperparameters from [Schmitt et al. 2020]. Table 1 shows sample predictions from these models. Table 2 compares their precision, recall, and F1 measures. T5 outperforms other models by a large margin.

Graphs from Images

The goal of this experiment is to evaluate the performance of scene graph classification after fine-tuning the pipeline using

	Method	R@50		R@100	
		1%	10%	1%	10%
SGCIs	R _{text→graph}	10.90 ± 0.12	24.96 ± 0.15	11.80 ± 0.11	26.09 ± 0.15
	SSGP	14.35 ± 0.15	26.11 ± 0.19	15.14 ± 0.17	27.12 ± 0.22
	CopyNet	14.46 ± 0.31	26.05 ± 0.29	15.19 ± 0.24	27.08 ± 0.26
	TXM - T5	14.53 ± 0.34	26.16 ± 0.32	15.28 ± 0.38	27.22 ± 0.28
	GT	14.72 ± 0.38	26.33 ± 0.45	15.36 ± 0.38	27.37 ± 0.47
PredCIs	R _{text→graph}	23.34 ± 0.10	49.99 ± 0.12	26.83 ± 0.15	54.40 ± 0.12
	SSGP	54.65 ± 0.14	55.65 ± 0.15	59.33 ± 0.18	59.67 ± 0.20
	CopyNet	56.24 ± 0.31	59.27 ± 0.28	60.35 ± 0.20	63.28 ± 0.25
	TXM - T5	58.64 ± 0.34	59.31 ± 0.30	63.07 ± 0.37	63.32 ± 0.24
	GT	62.02 ± 0.10	61.71 ± 0.19	65.68 ± 0.12	65.42 ± 0.19

Table 3: SGCIs and PredCIs results using different text-to-graph modules. We have substituted the missing 99% and 90% of annotated images with the textual knowledge extracted from their scene descriptions.

textual knowledge only. We evaluate our models for object classification, predicate classification (PredCIs - predicting predicate labels given a ground truth set of object boxes and object labels) and scene graph classification (SGCIs - predicting object and predicate labels, given the set of object boxes) on the test sets. Since the focus of our study is not improving object detection, we skip scene graph detection. Our ablation study concerns the following configurations:

- **SPB**: In this setting, both the backbone and the relational reasoning component are trained by *supervised learning* on the IM and SGs (1% or 10%) from the parallel set.
- **SCH**: Here, the backbone is trained by *self-supervised learning* on all VG images (without labels), and the relational reasoning component is trained on the IM and SGs (1% or 10%) from the parallel set.
- **TXM**: Here, the backbone is trained by *self-supervised learning* on all VG images (without labels), and the relational reasoning component is trained on the IM and SGs (1% or 10%) from the parallel set and fine-tuned from the SGs predicted from the text set (99% or 90%) using the text-to-graph module.
- **GT**: Here, the backbone is trained by *self-supervised learning* on all VG images (without labels), and the relational reasoning component is trained on the IM and SGs (1% or 10%) from the parallel set, and fine-tuned from the *ground truth graphs* (99% or 90%), instead of the text-to-graph predictions.
- **FSPB**: Here, both the backbone and the relational reasoning component are trained by *supervised learning* on 100% of the VG annotated images. Meaning that we have redefined the parallel set to include 100% of the VG training data and we do not need to substitute the images with the text set anymore. The goal of this setting is to compute the maximum accuracy that our model achieves, when we have all the annotated images with ground truth SGs, instead of using their textual scene descriptions. The results of this settings are not included as a separate bar so that the other

bars maintain a meaningful scale. Instead, they are written above each table.

We use the Recall@K (**R@K**) for a metric. R@K computes the mean prediction accuracy in each image given the top K predictions. For the complete set of results under constrained and unconstrained setups [Yu et al. 2017], and also with the Macro Recall [Sharifzadeh et al. 2021] (**mR@K**), refer to the supplementary materials.

Figure 3 presents the results of the ablation study. As shown, fine-tuning with textual scene descriptions improves the classification results under all settings (TXM), substituting a large proportion of the omitted images. Furthermore, the results even outperform FSPB under PredCIs (recall that the scene descriptions are sometimes complementary to image annotations and contain additional information).

Table 3 presents additional results also using different text-to-graph baselines. We can see that fine-tuning with the predicted graphs using T5, is as effective as fine-tuning with the crowd-sourced ground truth graphs (GT), and in some settings even better (object classification with 1%). Notice that compared to the self-supervised baseline, we gained up to $\sim 5\%$ *relative improvement* in object classification, more than $\sim 26\%$ in scene graph classification, and $\sim 31\%$ in predicate prediction accuracy. As expected, the choice of text-to-graph module has a larger effect on the PredCIs compared to the SGCIs and ObjCIs, due to the fact that SGCIs and ObjCIs rely heavily on the image-based features, whereas PredCIs has a strong dependency to relational knowledge. In supplementary materials we also provide additional results on the improvements per object class after fine-tuning the model with the textual knowledge (From SCH to TXM) and show that most improvements occur in under-represented classes. Figure 4 provides some qualitative examples of the predicted scene graphs before and after fine-tuning with the texts.

Note that while a significant stream of research works on the Visual Genome has been focused on utilizing 100% of the annotated image data and using a pre-trained VGG-16 [Simonyan and Zisserman 2014] backbone, in this work, we are focused on the few-shot learning setting and using a

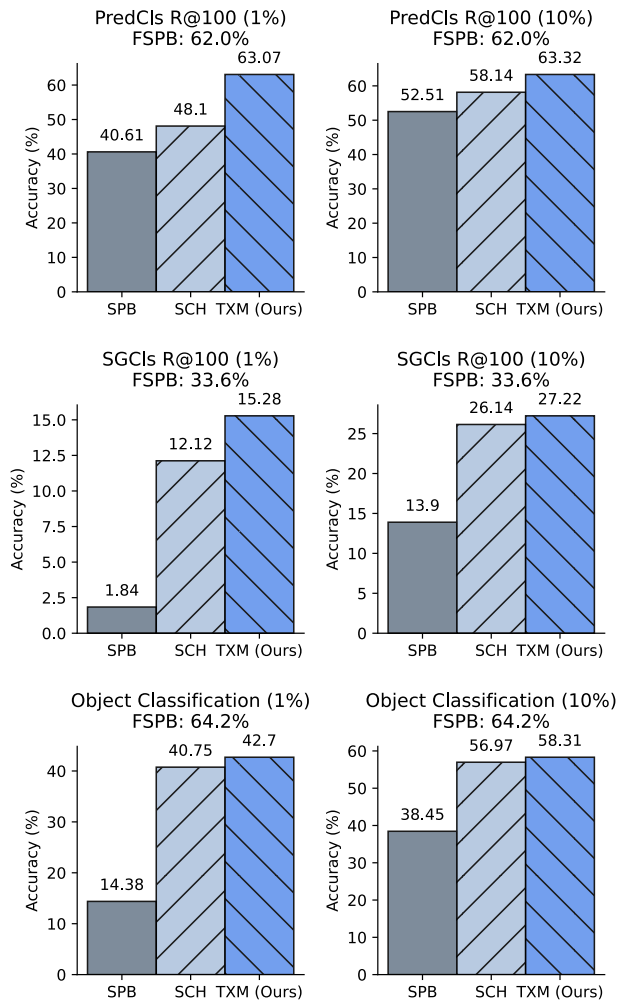


Figure 3: Fine-tuning with the textual knowledge (TXM) significantly improves the results in all settings of PredCls (top), SGCLs (middle), and object classification (bottom).

ResNet-50 [He et al. 2016] based self-supervised backbone (BYOL [Grill et al. 2020]). Nevertheless, to gain an intuition on our general performance, Table 4 present the results of our architecture using a VGG-16 Simonyan and Zisserman [2014], outperforming other works.

Conclusion

In this work, we proposed the first relational image-based classification pipeline that can be fine-tuned directly from the large corpora of unstructured knowledge available in texts. We generated structured graphs from textual input using different rule-based or embedding-based approaches. We then fine-tuned the relational reasoning component of our classification pipeline by employing the canonical representations of each entity in the generated graphs. We showed that we gain a significant improvement in all settings after employing the generated knowledge within the classification pipeline. In most cases, the accuracy was similar to when using the

Method	SGCLs		PredCls	
	R@50	R@100	R@50	R@100
VRD [Lu et al. 2016]	11.8	14.1	27.9	35.0
IMP+ [Xu et al. 2017]	34.6	35.4	59.3	61.3
SMN [Zellers et al. 2018]	35.8	36.5	65.2	67.1
KERN [Chen et al. 2019c]	36.7	37.4	65.8	67.6
VCtree [Tang et al. 2019]	38.1	38.8	66.4	68.1
CMAT [Chen et al. 2019a]	39.0	39.8	66.4	68.1
SIG [Wang et al. 2020]	36.6	37.3	66.3	68.1
GB-Net [Zareian et al. 2020]	38.0	38.8	66.6	68.2
TXM	39.0	39.9	66.7	68.3

Table 4: Comparing the general performance of the architecture to some other methods under the VG test set.

ground truth graphs that are manually annotated by crowd-sourcing.

References

- Ali, M.; Berrendorf, M.; Hoyt, C. T.; Vermue, L.; Galkin, M.; Sharifzadeh, S.; Fischer, A.; Tresp, V.; and Lehmann, J. 2020a. Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *arXiv preprint arXiv:2006.13365*.
- Ali, M.; Berrendorf, M.; Hoyt, C. T.; Vermue, L.; Sharifzadeh, S.; Tresp, V.; and Lehmann, J. 2020b. Pykeen 1.0: A python library for training and evaluating knowledge graph embeddings. *arXiv preprint arXiv:2007.14175*.
- Baier, S.; Ma, Y.; and Tresp, V. 2017. Improving visual relationship detection using semantic modeling of scene descriptions. In *International Semantic Web Conference*, 53–68. Springer.
- Baier, S.; Ma, Y.; and Tresp, V. 2018. Improving information extraction from images with learned semantic models. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 5214–5218. AAAI Press.
- Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, 2787–2795.
- Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. *Computing Research Repository*, arXiv:2005.14165.
- Chen, L.; Zhang, H.; Xiao, J.; He, X.; Pu, S.; and Chang, S.-F. 2019a. Counterfactual critic multi-agent training for scene graph generation. In *Proceedings of the IEEE International Conference on Computer Vision*, 4613–4623.
- Chen, T.; Xu, M.; Hui, X.; Wu, H.; and Lin, L. 2019b. Learning semantic-specific graph representation for multi-label

- Hajishirzi, H. 2019. Text generation from knowledge graphs with graph transformers. *arXiv preprint arXiv:1904.02342*.
- Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D. A.; et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1): 32–73.
- Lu, C.; Krishna, R.; Bernstein, M.; and Fei-Fei, L. 2016. Visual relationship detection with language priors. In *European Conference on Computer Vision*, 852–869. Springer.
- Nickel, M.; Murphy, K.; Tresp, V.; and Gabrilovich, E. 2016. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1): 11–33.
- Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.
- Ribeiro, L. F. R.; Schmitt, M.; Schütze, H.; and Gurevych, I. 2020. Investigating Pretrained Language Models for Graph-to-Text Generation. *Computing Research Repository*, arXiv:2007.08426.
- Santoro, A.; Raposo, D.; Barrett, D. G.; Malinowski, M.; Pascanu, R.; Battaglia, P.; and Lillicrap, T. 2017. A simple neural network module for relational reasoning. In *Advances in neural information processing systems*, 4967–4976.
- Schmitt, M.; Sharifzadeh, S.; Tresp, V.; and Schütze, H. 2020. An unsupervised joint system for text generation from knowledge graphs and semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 7117–7130.
- Schuster, S.; Krishna, R.; Chang, A.; Fei-Fei, L.; and Manning, C. D. 2015. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the fourth workshop on vision and language*, 70–80.
- Sharifzadeh, S.; Baharlou, S. M.; Berrendorf, M.; Koner, R.; and Tresp, V. 2021. Improving Visual Relation Detection using Depth Maps. In *2020 25th International Conference on Pattern Recognition (ICPR)*, 3597–3604.
- Sharifzadeh, S.; Baharlou, S. M.; and Tresp, V. 2021. Classification by Attention: Scene Graph Classification with Prior Knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 5025–5033.
- Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Singh, K. K.; Divvala, S.; Farhadi, A.; and Lee, Y. J. 2018. Dock: Detecting objects by transferring common-sense knowledge. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 492–508.
- Tang, K.; Zhang, H.; Wu, B.; Luo, W.; and Liu, W. 2019. Learning to compose dynamic tree structures for visual contexts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6619–6628.
- Tresp, V.; Sharifzadeh, S.; and Konopatzki, D. 2019. A Model for Perception and Memory.
- Tresp, V.; Sharifzadeh, S.; Konopatzki, D.; and Ma, Y. 2020. The Tensor Brain: Semantic Decoding for Perception and Memory. *arXiv preprint arXiv:2001.11027*.
- Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.
- Wang, W.; Wang, R.; Shan, S.; and Chen, X. 2020. Sketching image gist: Human-mimetic hierarchical scene graph generation. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, 222–239. Springer.
- Wang, X.; Ye, Y.; and Gupta, A. 2018. Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6857–6866.
- Wu, C.; Lenz, I.; and Saxena, A. 2014. Hierarchical Semantic Labeling for Task-Relevant RGB-D Perception. In *Robotics: Science and systems*.
- Xu, D.; Zhu, Y.; Choy, C. B.; and Fei-Fei, L. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5410–5419.
- Yaghoobzadeh, Y.; Adel, H.; and Schütze, H. 2017. Noise Mitigation for Neural Entity Typing and Relation Extraction. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 1183–1194. Valencia, Spain: Association for Computational Linguistics.
- Yang, J.; Lu, J.; Lee, S.; Batra, D.; and Parikh, D. 2018. Graph r-cnn for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 670–685.
- Yu, R.; Li, A.; Morariu, V. I.; and Davis, L. S. 2017. Visual relationship detection with internal and external linguistic knowledge distillation. In *IEEE International Conference on Computer Vision (ICCV)*.
- Zareian, A.; Karaman, S.; and Chang, S.-F. 2020. Bridging knowledge graphs to generate scene graphs. In *European Conference on Computer Vision*, 606–623. Springer.
- Zareian, A.; You, H.; Wang, Z.; and Chang, S.-F. 2020. Learning Visual Commonsense for Robust Scene Graph Generation. *arXiv preprint arXiv:2006.09623*.
- Zellers, R.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6720–6731.
- Zellers, R.; Yatskar, M.; Thomson, S.; and Choi, Y. 2018. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5831–5840.
- Zhang, H.; Kyaw, Z.; Chang, S.; and Chua, T. 2017. Visual Translation Embedding Network for Visual Relation Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017*, 3107–3115. IEEE Computer Society. ISBN 978-1-5386-0457-1.

Appendix

In this section, we present the implementation details and the statistical information of the VG splits that we have used for training and testing. Furthermore, we present more qualitative images and complementary quantitative results (constrained vs unconstrained, mR@K vs R@K). Finally, we present the results *per predicate* and *per object* category.

Implementation Details

We use ResNet-50 [He et al. 2016] for the backbone. We train the supervised backbones on the corresponding split of visual genome training set with the Adam optimizer [Kingma and Ba 2014] and a learning rate of 10^{-5} for 20 epochs with a batch size of 6. We train the self-supervised backbones with the BYOL [Grill et al. 2020] approach. We fine-tune the pre-trained *self-supervised* weights over ImageNet [Deng et al. 2009], on the entire training set of Visual Genome images in a self-supervised manner with no labels, for three epochs with a batch size of 6, SGD optimizer with a learning rate of 6×10^{-5} , the momentum of 0.9 and weight decay of 4×10^{-4} . Similar to BYOL, we use an MLP hidden size of 512 and a projection layer size of 128 neurons. Then for each corresponding split, we fine-tune the weights in a supervised manner with the Adam optimizer and a learning rate of 10^{-5} for four epochs with a batch size of 6.

After extracting the image-based embeddings from the penultimate fully connected (fc) layer of the backbones, we feed them to an fc-layer with 512 neurons and a Leaky ReLU with a slope of 0.2, together with a dropout rate of 0.8. This gives us initial object node embeddings. We apply an fc-layer with 512 neurons and Leaky ReLU with a slope of 0.2 and dropout rate of 0.1 to the extracted spatial vector \mathbf{t} to initialize predicate embeddings. We take four graph transformer layers of 5 heads for the relational reasoning component, each with 2048 and 512 neurons in each fully connected layer. We initialize the layers using Glorot weights [Glorot and Bengio 2010]. We train our supervised models with the Adam optimizer and a learning rate of 10^{-5} and a batch size of 22, with five epochs for the 1% and 11 epochs for 10%. We train our self-supervised model with the Adam optimizer and a

Total number of	1%	10%	100%
Images	577	5772	57723
Triples	4115.7 ± 51.0	40382.2 ± 181.4	405860
Objects	150	150	150
Predicates	45.25 ± 0.43	49.50 ± 0.5	50

Table 1: The statistics of the *parallel* set and the entire training set. The rows for triples and predicates indicate the mean and variance calculated over four different random splits.

learning rate of 10^{-5} and a batch size of 22, with six epochs for the 1% and 11 epochs for 10%. Finally, we train our self-supervised model, including the textual knowledge with the Adam optimizer and a learning rate of 10^{-5} with a batch size of 16, with six epochs for the 1% and 11 epochs for 10%. We use a Geforce RTX 2080 for our experiments.

Dataset Details

We use the sanitized version [Xu et al. 2017] of Visual Genome dataset which contains scene images with their corresponding scene graphs and textual descriptions. We use 57,723 samples as the full (100%) training set, 5,000 samples as our validation set, and 26,446 samples as our test set. As discussed in the paper, we randomly sample 1% and 10% of the entire training data to create the *parallel* set. The text set contains the remaining data (90% and 99%). The statistics of the data are shown in Table 1. Each split (1% or 10%) has been sampled randomly four times while keeping the number of images constant. The number of triples and predicates in Table 1 indicate the mean and variance over the four random samples.

Additional Qualitative Results

Figure 3 present qualitative results of the generated scene graphs from the VG test set and using our model (after fine-tuning it with the textual data).

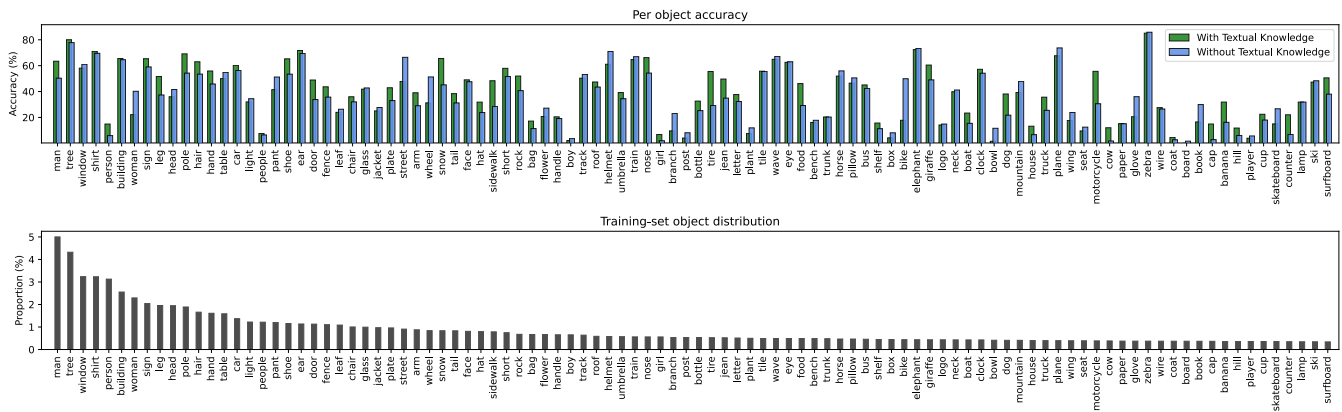


Figure 1: This plot shows per object classification improvement (Recall@ 100) before and after fine-tuning our model with textual knowledge with 1% split, the bottom plot shows the sample proportion for the objects in Visual Genome training set.

Per Predicate and Per Object Improvements

Figure 1 and 2 presents bar plots representing per object and per predicate improvement in accuracy, after fine-tuning with the texts. It also presents the frequency of their appearance in the training set. The goal of this table is to understand better where the improvements are happening; the results indicate that most improvements occur in under-represented classes. This means that we have achieved a generalization performance beyond the simple reflection of the dataset’s statistical bias. For example, interestingly we have improved the classification of objects such as a *Motorcycle* and *Surfboard* even though they only occur a few times in the training set.

Complementary Quantitative Results

Table 2-6 present complementary quantitative results including the results under *constrained* and *unconstrained* setups [Yu et al. 2017]. In the unconstrained setup, we allow for multiple predicate labels, whereas we only take the top-1 predicted predicate label in the constrained setup. Additionally, since the distribution of labeled relations in the Visual Genome is highly imbalanced, we also report the results with the Macro Recall [Sharifzadeh et al. 2021, Chen et al. 2019c] (**mR@K**) metric; **mR@K** reflects the improvements in the long tail of the distribution by taking the mean over recall *per predicate*.

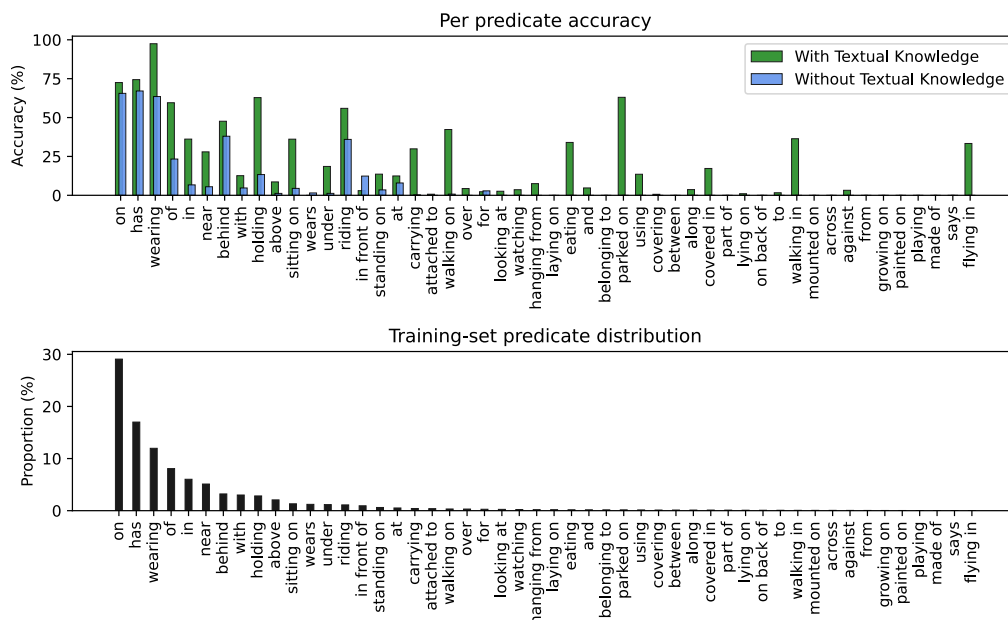


Figure 2: This plot shows per predicate classification improvement (Recall@ 100) before and after fine-tuning our model with textual knowledge with 1% split, the bottom plot shows the sample proportion for the predicates in Visual Genome training set.

	Method	Source	R@50		R@100	
			1%	10%	1%	10%
SGCls	SPB	Images	1.98 ± 0.28	16.24 ± 1.13	2.40 ± 0.30	18.15 ± 1.23
	SCH	Images	13.34 ± 0.51	30.45 ± 0.95	15.75 ± 0.64	33.94 ± 0.94
	TXM - T5	Images + Text	17.14 ± 0.37	31.43 ± 0.46	19.42 ± 0.47	35.03 ± 0.49
	GT	Images + GT	17.54 ± 0.43	31.72 ± 0.61	19.59 ± 0.49	35.33 ± 0.65
PredCls	SPB	Images	42.46 ± 0.99	60.21 ± 0.93	55.08 ± 1.10	71.87 ± 0.84
	SCH	Images	51.77 ± 0.59	65.89 ± 0.42	63.64 ± 0.75	76.85 ± 0.38
	TXM - T5	Images + Text	68.05 ± 0.21	69.74 ± 0.20	79.17 ± 0.16	80.54 ± 0.10
	GT	Images + GT	73.89 ± 0.19	73.25 ± 0.23	84.30 ± 0.16	83.69 ± 0.20

Table 2: Comparison of R@50 and R@100 with no graph constraints, for SGCls and PredCls tasks.

	Method	Source	mR@50		mR@100	
			1%	10%	1%	10%
SGCls	SPB	Images	0.29 ± 0.07	4.65 ± 0.62	0.43 ± 0.10	6.72 ± 0.77
	SCH	Images	2.54 ± 0.42	9.93 ± 0.35	3.71 ± 0.49	13.89 ± 0.51
	TXM - T5	Images + Text	4.31 ± 0.35	9.77 ± 0.35	6.25 ± 0.38	13.75 ± 0.52
	GT	Images + GT	4.84 ± 0.13	9.94 ± 0.54	6.83 ± 0.16	14.11 ± 0.64
PredCls	SPB	Images	5.89 ± 0.26	15.02 ± 1.37	9.62 ± 0.38	23.22 ± 1.78
	SCH	Images	9.24 ± 1.31	20.52 ± 0.53	14.28 ± 1.63	30.24 ± 0.72
	TXM - T5	Images + Text	23.72 ± 0.28	24.37 ± 0.72	35.72 ± 0.37	36.28 ± 0.80
	GT	Images + GT	27.11 ± 0.44	25.76 ± 0.74	40.35 ± 0.77	38.48 ± 0.82

Table 3: Comparison of mR@50 and mR@100 with no graph constraints, for SGCls and PredCls tasks.

	Method	Source	R@50		R@100	
			1%	10%	1%	10%
SGCls	SPB	Images	1.65 ± 0.26	13.37 ± 0.94	1.84 ± 0.26	13.90 ± 0.97
	SCH	Images	11.19 ± 0.41	25.16 ± 0.79	12.12 ± 0.47	26.14 ± 0.77
	TXM - T5	Images + Text	14.53 ± 0.34	26.16 ± 0.32	15.28 ± 0.38	27.22 ± 0.28
	GT	Images + GT	14.72 ± 0.38	26.33 ± 0.45	15.36 ± 0.38	27.37 ± 0.47
PredCls	SPB	Images	34.92 ± 0.81	48.69 ± 1.24	40.61 ± 0.84	52.51 ± 1.19
	SCH	Images	43.13 ± 0.59	54.40 ± 0.39	48.10 ± 0.54	58.14 ± 0.35
	TXM - T5	Images + Text	58.64 ± 0.34	59.31 ± 0.30	63.07 ± 0.37	63.32 ± 0.24
	GT	Images + GT	62.02 ± 0.10	61.71 ± 0.19	65.68 ± 0.12	65.42 ± 0.19

Table 4: Comparison of R@50 and R@100 with graph constraints, for SGCls and PredCls tasks.

	Method	Source	mR@50		mR@100	
			1%	10%	1%	10%
SGCls	SPB	Images	0.19 ± 0.04	2.21 ± 0.34	0.22 ± 0.04	2.43 ± 0.36
	SCH	Images	1.53 ± 0.20	5.31 ± 0.39	1.71 ± 0.22	5.80 ± 0.40
	TXM - T5	Images + Text	2.48 ± 0.26	5.04 ± 0.23	2.73 ± 0.29	5.53 ± 0.27
	GT	Images + GT	2.45 ± 0.05	5.17 ± 0.23	2.68 ± 0.06	5.68 ± 0.23
PredCls	SPB	Images	3.80 ± 0.20	8.16 ± 0.67	4.56 ± 0.22	9.45 ± 0.76
	SCH	Images	5.64 ± 0.56	11.35 ± 0.34	6.61 ± 0.62	12.96 ± 0.32
	TXM - T5	Images + Text	15.71 ± 0.63	16.20 ± 0.60	19.42 ± 0.84	19.38 ± 0.48
	GT	Images + GT	14.80 ± 0.31	14.59 ± 0.60	17.06 ± 0.39	16.92 ± 0.58

Table 5: Comparison of mR@50 and mR@100 with graph constraints, for SGCls and PredCls tasks.

Method	Source	Object Classification	
		1%	10%
SPB	Images	14.38 ± 0.57	38.45 ± 1.21
SCH	Images	40.75 ± 0.48	56.97 ± 0.76
TXM - T5	Images + Text	42.70 ± 0.49	58.31 ± 0.45
GT	Images + GT	42.09 ± 0.65	58.60 ± 0.56

Table 6: Comparison of object classification accuracy (Top-1) on the provided VG splits.

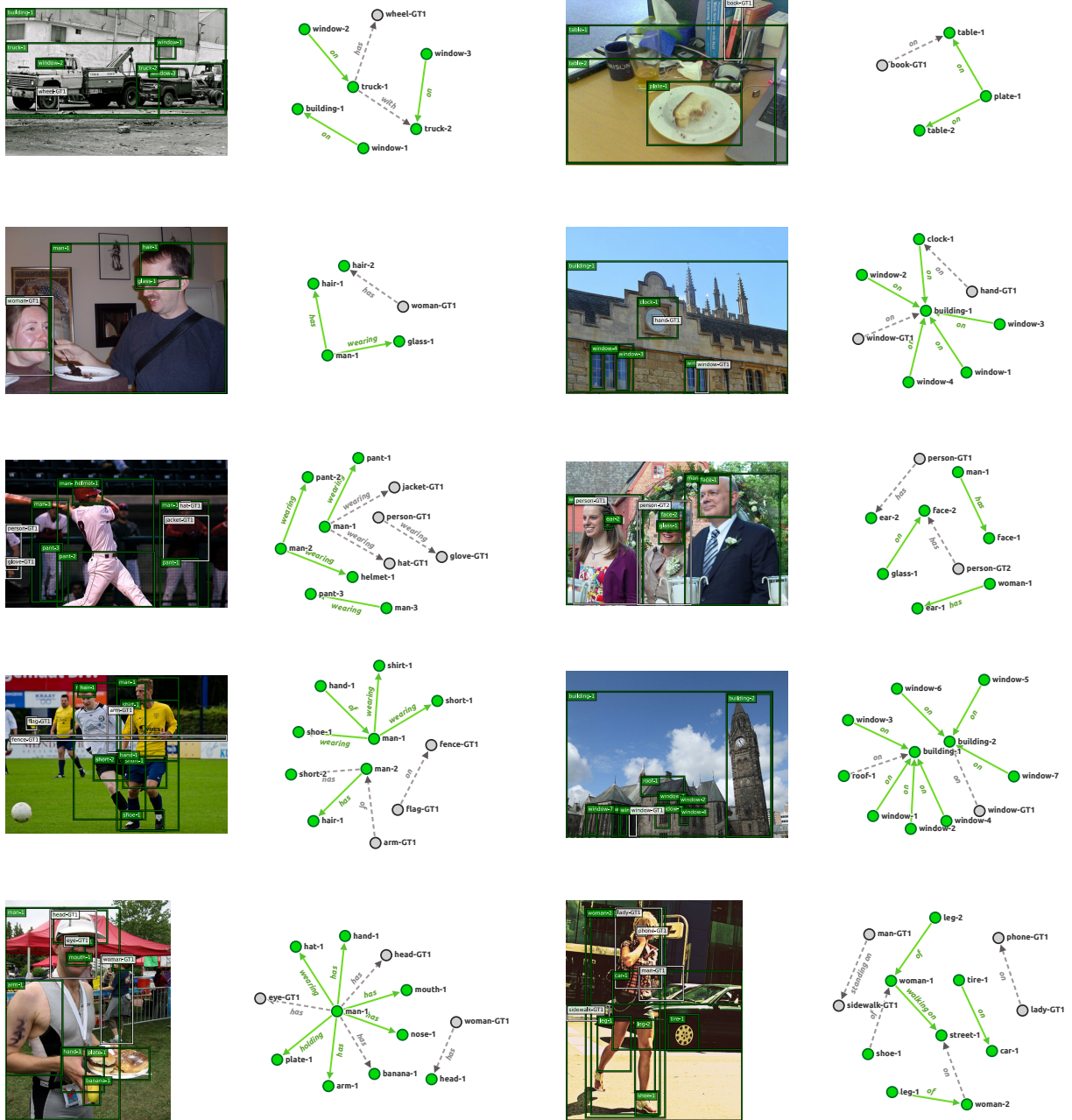


Figure 3: Qualitative examples of scene graph classification results (Recall@100) using the model trained with the textual knowledge (1% split). Green and gray colors indicate true positives and false negatives concluded by the model.