Beyond Manual Labels: Unsupervised Graph-Based Explanations for Error Analysis in Image Classifiers

Youmna Ismaeil[1,2][0000−0001−5663−6559], Jan-Hendrik Metzen ⋆3[0000−0002−6349−7806], Trung-Kien Tran[2][0000−0002−9938−1944], Hendrik Blockeel[1][0000−0003−0378−3699], and Daria Stepanova[2][0000−0001−8654−5121]

[1] KU Leuven, Leuven, Belgium
first.last@kuleuven.be
[2] Bosch Center for Artificial Intelligence, Renningen, Germany
first.last@de.bosch.com
[3] IPAI Aleph Alpha Research, Heidelberg, Germany
janhendrik.metzen@aleph-alpha-ip.ai

Abstract. Training datasets often contain subtle but irrelevant patterns that bias image classifiers and limit their generalization. Prior research has focused on detecting biases in misclassified data using manually defined dimensions, such as age or gender. However, since these approaches rely on manually predefined dimensions, they are labor-intensive and bound to be limited in terms of their coverage. Moreover, existing methods overlook biases present in correctly classified data. To address these issues, we propose an unsupervised framework that leverages commonsense knowledge graphs and open-source foundation models to automatically derive semantic dimensions and their values, identifying biases that influence correct and incorrect classifications of data. Using these dimensions and values, we construct scene graphs and identify distinctive graph patterns for correctly and incorrectly classified data, uncovering how combinations of semantic dimensions impact classifier decisions. Evaluations confirm that our scene graphs are of high quality, as they include information from manual annotations while being denser and more informative than manually constructed graphs. Moreover, our framework demonstrates high predictive accuracy, effectively identifying patterns responsible for correct and incorrect classifications, with F1 scores ranging from 0.72 to 0.96. It also proves effective for error analysis and proactive bias detection in datasets.

Keywords: Explainable AI · Scene Graphs · Bias Detection · Knowledge Graphs

1 Introduction

Neural networks achieve impressive classification performance through high-quality training data, but their ability to generalize weakens when the data is biased. For instance, a classifier trained on classrooms with green desks may fail to generalize to other desk colors. Such biases are prevalent in datasets like CelebA [29], where blond

⋆ This work was done while at Bosch Center for Artificial Intelligence

females dominate certain labels. Exploring and understanding failures of image classifiers is challenging and has been addressed in several works [26,38,4,22,31,16]. These studies typically rely on manually predefined dimensions (e.g., age or gender), which are labor-intensive and restrict bias analysis to fixed dimensions and failure cases only that may not encompass the full spectrum of possible bias causes.

To address these shortcomings, we propose an unsupervised framework that combines commonsense knowledge graphs (CSKGs), foundation models, and graph mining techniques to extract patterns from automatically extracted dimensions, uncovering learned bias. This approach enables the discovery of complex patterns that go beyond limited predefined labels by generating detailed, task-agnostic scene descriptions. These descriptions include object attributes (e.g., shape, color), environmental settings (e.g., dark, cloudy, rainy), and visual styles (e.g., blurry, high or low contrast), addressing gaps left by previous scene description methods [32,52,47,5]. The descriptions are structured as scene graphs and analyzed using graph mining techniques to uncover patterns that emulate classifier behavior, revealing how input components influence predictions. E.g., a pattern can state that images correctly classified as classrooms often feature green desks and wooden chairs; at the same time, another pattern might indicate that incorrectly classified images typically depict blue desks and metal chairs.

Additionally, we utilize insights from the CSKG and extracted patterns to predict potential failures, enhancing the robustness of image classifiers and helping avoid failures due to low or no coverage of specific scenarios in the training data. For example, if a pattern shows that classrooms with blue desks and metal chairs are often misclassified, collecting more such images for retraining could improve classifier performance. We also make use of the extracted patterns to help in understanding and justifying failures made by an image classifier. The main contributions of this work are:

- An unsupervised framework that extracts **graph-based patterns** from automatically extracted semantic dimensions, enabling the discovery of complex success and failure patterns learned by an image classifier.
- An unsupervised scene descriptor that uniquely utilizes CSKGs and foundation models to create detailed scene graphs for images.
- A novel use of sub-graph mining to extract patterns for analyzing failure cases in image classifiers, providing structured insights into their behavior.
- A novel application of the framework to predict and explain failure cases using structured graph explanations for systematic error analysis.

2 Framework

Before delving into the details of our framework, we introduce the notations used throughout this work.

- A Knowledge Graph (KG) denoted as $\mathcal{KG}$ comprises triples in the form (s, r, o), where s (subject) and o (object) are entities or concepts and r denotes the relationship between them. For example, the triple *(desk, is, blue)* indicates that the desk is blue. In this work, we utilize ConceptNet 5.5 [40], a commonsense knowledge

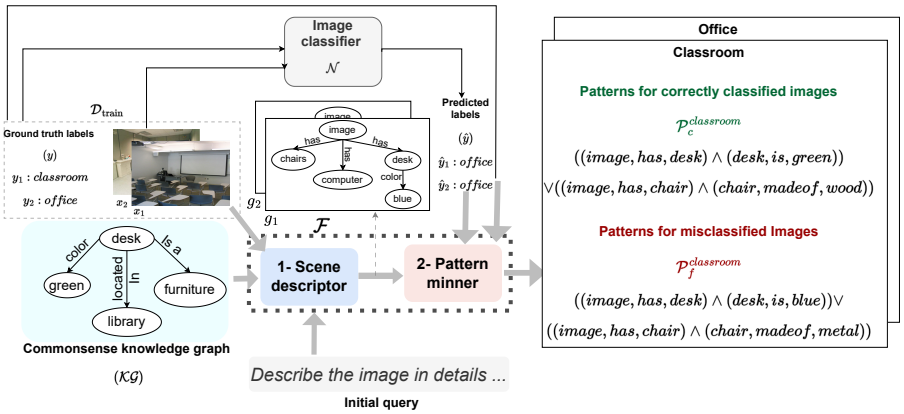

Fig. 1: Illustration of the proposed framework for pattern extraction. The framework features an unsupervised scene descriptor for generating scene graphs, and a pattern miner that identifies common graph patterns among scene graphs for correctly and misclassified images per class.

graph known for its high coverage and open access, though other CSKGs can also be used in our framework.

- $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{m}$ represents the dataset composed of image-label pairs, where x_i is the i^{th} image, y_i its corresponding class label and m denotes the total number of images. The dataset is divided into 3 subsets: training ($\mathcal{D}_{\text{train}}$), validation ($\mathcal{D}_{\text{valid}}$), and testing ($\mathcal{D}_{\text{test}}$).
- $\mathcal{N}$ denotes the image classifier trained on $\mathcal{D}_{\text{train}}$ to predict the label $\hat{y}$ from the input image x.

Our framework $\mathcal{F}$, shown in Fig. 1, takes $\mathcal{KG}$ and $\mathcal{D}_{\text{train}}$ as input and outputs patterns reflecting distinguishing commonalities among correctly classified images (w.r.t. falsely classified ones), and separately, patterns for falsely classified images (w.r.t. correctly classified images) for each class k. To extract these patterns for each class, our framework includes two main modules: a scene descriptor that gathers detailed information from images in $\mathcal{D}_{\text{train}}$, and a pattern extractor that identifies recurring patterns in the scene descriptions. In the following sections, we explain our framework in detail.

2.1 Scene Descriptor

The scene descriptor (see Fig. 2) takes the images $\{x_1, \ldots, x_q\}$ from $\mathcal{D}_{\text{train}}$, and a CSKG $\mathcal{KG}$ as input and outputs a set of graphs $\{g_1, \ldots, g_q\}$, with each graph thoroughly describing a single image. To generate these graphs, we propose to rely on a foundation model. In principle, any foundational model suitable for visual and textual question answering could be suited for this task. In our experiments we relied on the open-source LLaVA v1.5 model [27], which is pre-trained for visual and textual question answering, allowing us to generate scene descriptions without additional fine-tuning.

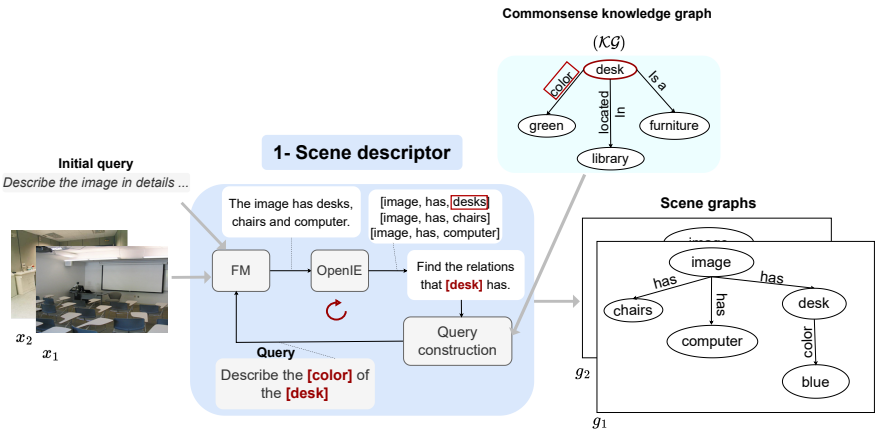

Fig. 2: The scene descriptor leverages $\mathcal{KG}$ and foundation models (FM) to generate scene graphs g_i for images x_i in an unsupervised fashion.

You are given an image, and your task is to write a paragraph describing the image in details. The description should be detailed enough to draw and paint the image again without seeing it. Provide reasoning why you think that the image has specific features. Answer in short, simple, full sentences. Stick to the image; do not mention information that is not in the image.

Describe the [relation] of the [node]. Your Answer has to be with respect to the image. If the object is not in the image or is not clear, skip the question. Answer in complete sentences. Do not answer with a single word. Do not answer with yes/no. Answer in short, simple, full sentences.

Fig. 3: On the left: details of the initial query (*"Describe the image in detail .."*) given to LLaVA for each image x_i. On the right: instructions given to LLaVA along with the question *"Describe the [relation] of the [node]."*

First, we prompt FM with an initial query asking it to generate a detailed description of a given image, including specific instructions on the desired level of detail, as outlined in Fig. 3. We then extract triplets from the foundation model's responses using OpenIE [1], an unsupervised method that does not require the fine-tuning typically needed by foundation models for triplet extraction [36]. Then we pass the triplets extracted from the answer of the initial query along with the $\mathcal{KG}$ to the query construction module that identifies proper semantic dimensions that each concept might have to ask relevant questions about them. If a concept is not in the CSKG, we extract the relations of its most similar concept based on embeddings computed using MiniLM [45], a task-agnostic, lightweight model. We treat the node's relation labels in the CSKG as semantic dimensions and then use the template *"Describe the [relation] of the [node]"*

as a query to the FM, along with the image, to extract visual insights about the node and its attributes.

For example, if *writing desk* is one of the nodes in the triplets extracted from the answer to the initial query, we search for a similar node to *writing desk* in the CSKG. Based on the embedding of *writing desk*, the closest node in the CSKG is *desk*, which is associated with several semantic dimensions, such as *color*. Leveraging this relationship, we query the FM about the *color* of the *desk* depicted in the image. This process is repeated for each node in the triplets extracted from the initial query's answer for image x_i, enabling the extraction of additional triplets from the responses.

These triplets form a level in the output scene graph g_i for image x_i. The depth of the scene graph can be increased by increasing the number of iterations, n, which is a tunable parameter within the scene descriptor module. This iterative approach allows us to control the level of granularity of the scene graph describing each image.

2.2 Pattern Miner

Once we obtain the scene graphs for each image in the form of triplets, we extract the common sub-graph patterns among images that are correctly classified for each class as well as the falsely classified images. To achieve this, we use cgSpan [37], which efficiently extracts frequent closed subgraph patterns from a set of graphs and a node-to-label mapping, providing compact representations without redundancies. We use a node-to-label mapping to ensure that cgSpan finds patterns among semantically similar but differently named nodes, such as *desk* and *writing desk*. This allows it to identify sub-graph patterns involving nodes with different names that belong to the same class.

To construct this mapping, we compute node embeddings using MiniLM [45], a task-agnostic, lightweight embedding model, and consider nodes with a cosine similarity of at least 0.8 to be similar. To prepare the input for cgSpan, we group the scene graphs by class and split them into graphs for correct and incorrect classifications based on the ground truth labels vs predicted labels. For example, as shown in Fig. 4 for $k = classroom$, g_1 and g_3 belong to images that were correctly classified by $\mathcal{N}$ as classroom, while g_2 and g_4 are scene graphs for misclassified images.

For each class, we pass the group of graphs for correctly classified images, along with the node-to-label mapping, to cgSpan to extract the most frequent closed sub-graph patterns, denoted as $\mathcal{P}_c^k$. We repeat this process for the graphs of images with the ground truth label that were misclassified to obtain $\mathcal{P}_f^k$. Each set of patterns $\mathcal{P}_c^k$ is composed of a disjunction of patterns (i.e., $\mathcal{P}_c^k = p_1 \vee \ldots \vee p_j$), where each pattern p_j is a conjunction of triplets, as all the triplets in p_j are crucial for forming the pattern. The pattern can be composed of only one triplet, and the same applies to $\mathcal{P}_f^k$. We then exclude the common sub-graph patterns between $\mathcal{P}_f^k$ and $\mathcal{P}_c^k$ from both sets, as shared patterns are uninformative.

CgSpan also returns a frequency score f_j for each extracted pattern p_j that can be used to filter patterns. We consider patterns appearing in at least 50% of input image graphs, balancing quality and quantity while ensuring that no patterns are returned if the model is poorly trained or performing randomly. The parameter settings and runtime details are discussed in Sec. 3.4.

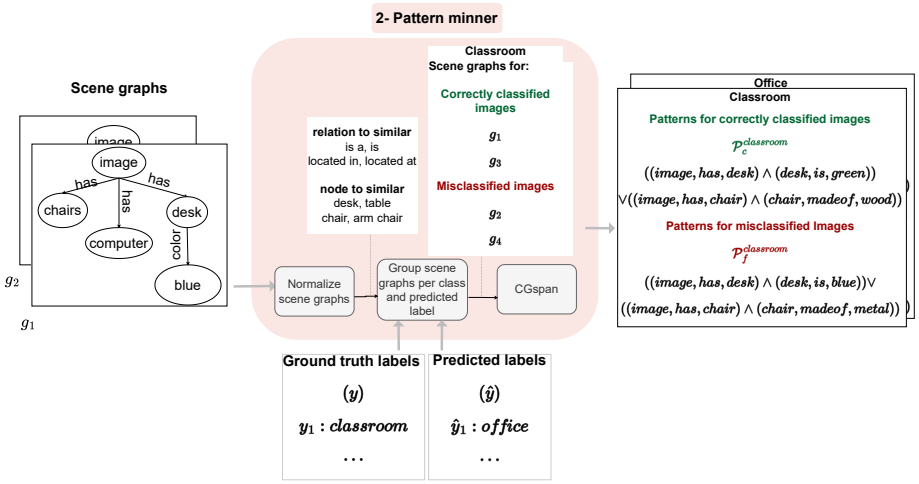

Fig. 4: The pattern miner extracts the common patterns between the scene graphs created by the scene descriptor.

3 Experiments and Results

In this section, we evaluate the scene graphs from the scene descriptor and patterns from the pattern miner using ResNet18 [15] as the classifier.

3.1 Scene Descriptor Evaluation

First, we evaluate the quality of scene graphs generated using our method by comparing them to those constructed based on manual annotations. In what follows, we set n, the parameter responsible for the number of iterations of the scene descriptor, to 2.

Dataset
We evaluate the scene descriptor using the *Visual Genome (VG)* dataset [23], which contains scene graphs created from manual annotations with an average of 35 objects, 26 attributes, and 21 relationships per image. For the evaluation of the scene graphs, we randomly select 100 images and their corresponding scene graphs.

Baseline
As a baseline for the scene descriptor, we utilized our framework with $n = 1$ and used the initial query *"Describe the image in details"*, without additional instructions. We use the following questions to assess the performance of the scene descriptor:

• Do scene graphs generated by the scene descriptor cover the aspects represented in scene graphs constructed based on human annotations?
This experiment evaluates graph quality by comparing the similarity between graphs generated by the scene descriptor and those derived from manual annotations. To handle lexical differences between the scene descriptor's output and the VG, we use embeddings for each triplet to measure similarity. Specifically, cosine similarity between

triplet embeddings from our scene graphs and those in the VG is calculated. For each image triplet in the VG, we identify the most similar triplet within the graph generated by our scene descriptor for the same image. The average similarity score is then computed using the formula defined in Eq. 1.

$$S(\hat{g}_i, g_i) = \frac{\sum_{\hat{\tau} \in \hat{g}_i} \max_{\tau \in g_i} (\text{Sim}(\xi(\tau), \xi(\hat{\tau})))}{|\hat{g}_i|} \tag{1}$$

where, $\hat{g}_i$ is the list of triplets for a given image within the VG dataset, while g_i represents the set of triplets for the same image generated by the scene descriptor. $\text{Sim}(\xi(\tau), \xi(\hat{\tau}))$ is the cosine similarity. The triplet embedding ξ is computed as the concatenation of embeddings: for triple $\tau = (s, r, t)$ we have $\xi(\tau) = [\text{Embedding}(s),$ $\text{Embedding}(r), \text{Embedding}(t)]$, where embeddings are computed using MiniLM. The same applies to $\xi(\hat{\tau})$. While triplet embeddings could in principle be derived from node embeddings generated by knowledge graph embedding models [2,13,43], these models generally focus on learning node embeddings from the graph structure and node neighborhoods, rather than the semantic meanings of the node labels.

Our findings show that the triplets in the VG dataset have comparable counterparts in the outputs generated by our scene descriptor, with an observed average similarity score of 0.79. This is noteworthy compared to the baseline similarity score of 0.48. These results validate the scene descriptor and its sub-components, LLaVA & OpenIE, in replicating the quality of manually annotated graphs and extracting high-quality triplets, showcasing its potential for detailed, unsupervised scene graph construction.

• Does the scene descriptor produce richer graphs than graphs obtained from manual annotations?

To investigate this, we compare the level of detail captured by our scene descriptor against the VG dataset. This is achieved by performing a comparative analysis focusing on the following five key components: triplets, relations, nodes, node clusters, and relation clusters. Our analysis evaluates the total number of nodes and relations as well as the diversity within these elements. We specifically examine cluster counts to determine whether variations in nodes (e.g., *desk* versus *writing desk*) or relation phrases (e.g., *has* versus *has a*) contribute to the differences in graph statistics. For that, we use HDBSCAN [30], which determines the optimal number of clusters based on data density, with a minimum cluster size of 2 and cluster selection $\epsilon = 0.1$.

Fig. 5 shows the overall number for each component. One can observe that graphs generated by our scene descriptor are denser compared to those from the VG dataset based on all five components. Further evidence is provided in Tab. 1 (with more examples from additional experimental results available in the supplementary material), where it is further confirmed that the VG graphs often contain less details about the scenes compared to the graphs generated by our scene descriptor. Indeed, as one can see in Tab. 1, the scene graphs constructed by our method contain such specific information as the dogs' color and species, the background, and the surfing board color in the 1^{st} row, as well as the towel color and the exact placement of the toilet and bathtub in the 2^{nd} row. These findings underscore the superior descriptive power and comprehensiveness of our scene descriptor, demonstrating its potential to enrich data in automated scene graph generation.

Image	**VG graph ($\hat{g}$)**	**Scene descriptor graph (g)**
	(boats, are on, ocean), (ship, on, water)	(fighter jets, flying in, formation), (fighter jets, flying in, coordinated pattern), (planes, synchronize, movement), (planes, are shaped like rockets), (planes, is, military), (altitude flyby, is, low), (primary, impressive aerial display of, military), (smoke, is coming from, planes), (smoke, is, white), (smoke, is, trailing), (smoke, high into, sky), (navy ship, are, multiple), (navy ship, are on, ocean), (navy ships, including, battleship), (scene, is, dynamic), (ocean, is in, background), (two cruise ships, are in, background), (island, is, background), (view, is, up-close), (event, is, outdoor), (exercises, are, military), . . .
	(dogs, have, shirts), (dogs, on, surfboard), (surfboard, on, water), (woman, in, water), (words, on, chest), (hat, on, head)	(dogs, are wearing, shirts), (dogs, wear, bandanas), (dogs, are, brown), (dogs, are on, surfboard), (dogs, are, labrador retrievers), (surfboard, is in, water), (surfboard, is, blue), (woman, is, old), (woman, is on, surfboard in water), (water, is, blue), (woman, wearing, pink hat), (custom, is with, reality rally), (scene, is, unusual), (day, is, sunny), (sky, is, clear), (apron, is, white), . . .
	(towel set, hanging on, holder), (tile, on, floor)	(towel, is, black), (towels, are, hung), (floor, is, black tiled), (natural light, enter, bathroom), (toilet, is, white), (toilet, is, in the corner against the wall), (toilet, is by, sink), (tub, under, window)

Table 1: VG graph vs. our scene descriptor graph for the same image, showcasing the richer details captured by our method.

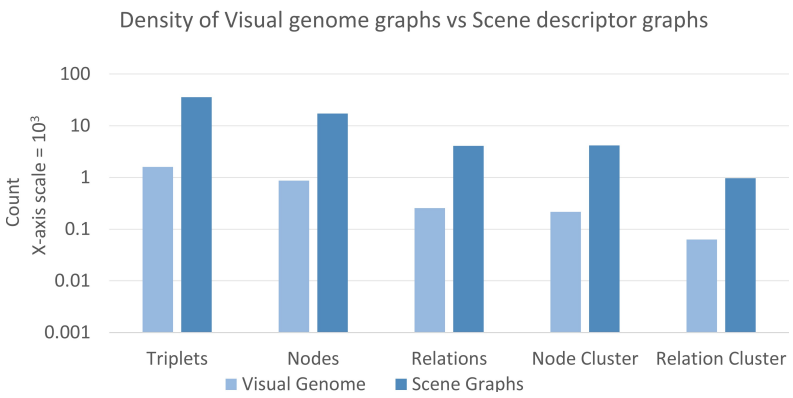

Fig. 5: A plot in logarithmic scale reflecting the total number of triplets, nodes, relations, node clusters, and relation clusters found in VG graphs vs. our scene graphs ($n = 2$).

3.2 Pattern Evaluation

In this section, we evaluate the quality of the extracted patterns using datasets with (1) known patterns and (2) unknown patterns. Our experiments aim to verify the framework's accuracy in detecting patterns recognized by image classifiers and to assess the quality of patterns in datasets lacking predefined patterns. In the experiments evaluating the pattern miner, the parameter n of our framework is set to 2 unless stated otherwise.

Datasets

For evaluation, we utilize the following datasets:

• **Biased CelebA**

CelebA [29] is a large facial attributes dataset annotated with 40 attributes. Following [35], we constructed Biased CelebA with 1000 *female* images with *blond hair* and 1000 *male* images with *dark hair*. The data was split in a 3:2 ratio for training and validation. For testing, we selected 2000 additional images with swapped attributes, where all *females* have *dark hair* and all *males* have *blond hair*.

• **Waterbirds**

The Waterbirds dataset [35] classifies images into *waterbird* and *landbird* categories. *Waterbirds* typically appear in images with *aquatic* backgrounds, and *landbirds* in images with *terrestrial* backgrounds respectively. We selected 500 images per class, splitting the 1000 images in 3:2 for training and validation. For testing, we chose 1000 additional images where the background context is swapped—*landbirds* have *aquatic* backgrounds, while *waterbirds* have *terrestrial* backgrounds.

• **MIT Indoor**

To challenge the classifier, we selected six visually similar classes (*bakery, deli, classroom, office, kitchen, and pantry*) from the MIT Indoor dataset [34]. This subset includes 2,662 images, stratified into a 3:2 ratio for training and testing.

Model

For the image classifier, we trained a ResNet18 model with parameter tuning similar to [35] for the CelebA and Waterbirds datasets. Evaluation was conducted on the val-

Dataset	training F1	validation F1	test F1
CelebA	0.71	0.71	0.52
Waterbirds	0.76	0.77	0.47
Indoor	0.55	-	0.49

Table 2: F1 scores of ResNet18 trained from scratch on various datasets.

Dataset	Class	Patterns from correct predictions ($\mathcal{P}_c^k$)
Biased CelebA	female	(fair, is, complexion) $\vee$ (hair, is with, blonde)
	male	(hair, is, darkness)
Waterbirds	landbird	((small, is, bird) $\wedge$ (bird, has, position in branch) $\wedge$ (bird, is focus of, image)$\wedge$ (trees, in, image)) $\vee$ ((background, is in, forest)$\wedge$ (bird, has, bright coloration), ...
	waterbird	(bird, is, seagull) $\vee$ ((bird, in, image) $\wedge$ (bird, on, water) $\wedge$ (large, is, bird) $\wedge$ (bird, is, black),...)

Table 3: Patterns extracted from the ResNet18 trained on the biased CelebA and Waterbirds datasets, sorted by frequency.

idation split, which exhibits patterns different from those present in the training split, to ensure the model captures dataset patterns. This is verified if the model trained on a biased data split performs better on a validation split with the same bias than on a test split with a different bias. For the MIT Indoor dataset, the model was trained for 20 epochs with a learning rate of 0.001. As the patterns in this dataset are unknown, we report F1 scores on the training and test splits.

Baseline

Given the novelty and fundamental difference of our structured graph pattern-based explanations from explanations provided in prior methods (see Section 4 for discussion), which rely on single-label explanations or visual illustrations, direct comparisons with earlier proposed methods is infeasible. Therefore, as a baseline, we use our framework with $n = 1$, i.e., we do not utilize the iterations to extract more information about the images, which also means that we do not use the CSKG to create scene graphs.

Relying on the introduced datasets and models we address the following questions:

• Can our framework identify known pattern bias in an image classifier?

To answer this question, we assess the quality of the patterns extracted by our framework both qualitatively and quantitatively.

Qualitative evaluation

Table 2 shows the F1 score of ResNet18 on training and validation splits that share the same bias, as well as on a test set with a different bias. The performance of ResNet18 on the CelebA and Waterbirds biased data sets indicates its capacity to capture specific patterns. This is evidenced by high F1 scores in the training and validation splits, which share the same patterns, but lower performance on test splits that present different patterns. Upon achieving this, we extract and compare patterns from the classifier's correct and incorrect predictions against known dataset biases. For example, the bias detected in the patterns extracted from ResNet18 trained on the CelebA dataset (Tab. 3) align

with its lack of generalization on the CelebA test set (Tab. 2) and the known patterns in the biased dataset [35], confirming the effectiveness of our framework in identifying biases contributing to classifier errors. This procedure was similarly applied to the Waterbirds dataset, with findings (Tab. 3) aligning with those observed in the biased CelebA dataset.

Quantitative evaluation

When the dataset patterns learned by the classification neural network $\mathcal{N}$ are unknown, we evaluate the effectiveness of the extracted patterns in replicating $\mathcal{N}$'s predictive behavior. This assessment is carried out using a pattern-based classifier, denoted as PC. The classifier PC operates by taking a scene graph g_i constructed by our scene descriptor and the patterns mined by the pattern miner for different classes. It then predicts a class $\dot{y}_i$ for each image x_i by evaluating the similarity between the triplets in the scene graph g_i for x_i and the extracted patterns for each class $\mathcal{P}_j^k$, where j can be c (correctly classified) or f (falsely classified), and $\mathcal{P}_c^k$ represents the set of patterns associated with images correctly classified under the class k. More formally, PC relies on the scores computed by Eq. 2 to predict the class k of an image x_i with scene graph g_i.

$$score_{j^k}(g_i, \mathcal{P}_j^k) = \frac{\sum_{p_a \in \mathcal{P}_j^k} f_a \cdot S(g_i, p_a)}{|\mathcal{P}_j^k|} \tag{2}$$

where p_a is a subgraph pattern, f_a is the cgSpan frequency score of p_a, and $S(g_i, p_a)$ is the average similarity between the scene graph g_i and the sub-graph pattern p_a (see Eq. 1). For convenience, in Eq. 2 each pattern $\mathcal{P}$ is treated as a set of conjunctions and a conjunction p as a set of triplets. After computing $score_{j^k}$ for $j \in \{c, f\}$ for each image x_i and each class k, we apply the softmax function to these scores to obtain the probability of each $\mathcal{P}_j^k$ and select the pattern with the highest probability. For example, if $\mathcal{P}_c^{office}$ has the highest probability, then the PC classifier outputs $office$. Following prior work [19], we compute the alignment score between the predictions of PC and the predictions of $\mathcal{N}$. However, unlike previous work, we use a weighted F1 score instead of percentage to better account for the varying number of images correctly and incorrectly classified by the neural network $\mathcal{N}$.

The weighted F1 alignment score measures the coherence between the predictions made by $\mathcal{N}$ and PC with respect to the ground truth labels. For each image x_i, let y_i be the ground truth label, $\hat{y}_i$ the prediction made by $\mathcal{N}$, and $\dot{y}_i$ the prediction by PC. The alignment score is computed as the weighted F1 score between two binary sequences: one indicating whether each $\hat{y}_i$ matches y_i (1 if $\hat{y}_i = y_i$, 0 otherwise), and the other indicating whether each $\dot{y}_i$ matches y_i (1 if $\dot{y}_i = y_i$, 0 otherwise). The score is the highest when both models make correct predictions ($\hat{y}_i = \dot{y}_i = y_i$) or both make the same error ($\hat{y}_i \neq y_i$ and $\dot{y}_i \neq y_i$).

Tab. 4 presents the weighted F1 alignment scores for patterns extracted using different frameworks. The results indicate that patterns generated by our framework outperform those from both the baseline and the FM-only variant, where CSKG is replaced by FM for dimensions extraction. These patterns more effectively and reliably predict whether $\mathcal{N}$ will correctly classify or misclassify an unseen image from the test set. This highlights the importance of incorporating CSKGs with iterative refinement in gener-

Dataset	Baseline	FM Only PC	(CSKG & FM) - PC (Ours)
Biased CelebA	0.64	0.66	0.80
Waterbirds	0.90	0.93	0.96
Indoor	0.67	0.63	0.72

Table 4: Weighted F1 alignment scores comparing $\mathcal{N}$'s predictions on the test set with labels generated by three variants of PC: (1) a baseline model, (2) an FM-only model that replaces CSKG with FM for attribute extraction, and (3) our full framework that combines CSKG and FM (with $n = 2$).

n	Waterbirds	CelebA	Indoor
1	0.90	0.64	0.67
2	0.96	0.80	0.72

Table 5: Weighted F1 alignment score for different values of n for different datasets.

ating informative scene graphs, which, in turn, enables the extraction of patterns that more closely align with $\mathcal{N}$'s predictions.

To assess whether the observed weighted F1 alignment scores could have occurred by chance, we performed a bootstrapped statistical significance test [11] comparing the weighted F1 scores of our proposed PC (using CSKG & FM) and $\mathcal{N}$. Bootstrapping was chosen for its flexibility and robustness in estimating p-values. Under the null hypothesis—assuming that the observed agreement between the two models' predictions is no greater than what would be expected by chance alone—the resulting p-values were found to be less than 0.001 across all datasets. This provides strong evidence that the alignment between the models is statistically significant and not attributable to random variation.

In Tab. 5, we present the results reflecting the effect of the parameter n on the granularity of the constructed scene graphs. We observe that larger values of n naturally result in more detailed graphs and yield better F1 alignment scores with the image classifier $\mathcal{N}$. When evaluated on the CelebA and Indoor datasets, our method benefits from two iterations of the scene graph computation, which provide richer information about the considered scene. In contrast, a single iteration is sufficient to achieve a weighted F1 alignment score of 0.9 on the simpler Waterbirds dataset. The impact of increasing the number of iterations n depends on the dataset complexity.

• Can the CSKG in our framework be replaced by a foundation model?
To address this question, we replaced the CSKG with a foundation model (LLaVA). Instead of using the CSKG to extract the respective dimensions, we instructed the LLaVA model to propose possible dimensions for an object by using the following prompt "*Return the list of properties that [node] has.*" Subsequently, we inquired about the values of the returned properties one by one. We call this variant of PC FM only PC. The results in Tab. 4 show that our framework, which utilizes CSKG for extracting semantic dimensions, is more effective in identifying patterns that mirror the behavior of $\mathcal{N}$

Fig. 6: Left: Top 3 images from the MIT Indoor dataset, manually labeled as *deli* but misclassified as *bakery* due to red-highlighted bakeries found in the extracted graph patterns for both the correct patterns of the *bakery* class and the false patterns of the *deli* class. Bottom 3 images are *deli* examples misclassified as *bakery*, due to the frequently occurring display cases and baked goods, identified by the graph-patterns in the false patterns of *deli* and correct patterns of *bakery*. Right: For reference, 6 *bakery* images from the same dataset, manually labeled as *bakery*. One can note the visual similarity to the *deli* images on the left, further demonstrating the validity and effectiveness of the insights uncovered by our patterns.

than the variant that replaces the CSKG with a foundation model. These results are consistent with prior findings [32,17,42,51,53], which show that multimodal foundation models often struggle with deep structured visual understanding when used alone, frequently overlooking objects' dimensions.

3.3 Use Cases

Our framework's extracted patterns have versatile applications. We highlight two: explaining the image classifier's errors and proactively detecting bias in datasets.

Error analysis
Confusion matrices are useful to understand which classes are often confused with each other, but they do not explain *why* this happens. Our framework can address this limitation. When an image classifier frequently misclassifies instances of class A as class B, patterns in $(\mathcal{P}_c^A \cap \mathcal{P}_f^B)$ reveal likely reasons. For instance, in the MIT Indoor dataset, the confusion matrix in Fig. 7 shows that *deli* is misclassified as *bakery* 56% of the time. Our framework reveals that the pattern *(goods, are, baked)*$\vee$*(items, in, display case)* appears both in correctly classified images of *bakeries* and in images misclassified as *delis*. This suggests that *delis*, like *bakeries*, have baked goods; or perhaps there are images of *bakeries* that are mistakenly labeled as *delis* in the training dataset.

Looking at some actual images reveals that both scenarios hold: *deli* images often have bread, as highlighted in red in the first row of *deli* images in Fig. 6. There are annotation errors within the training dataset, as highlighted in the second row of images

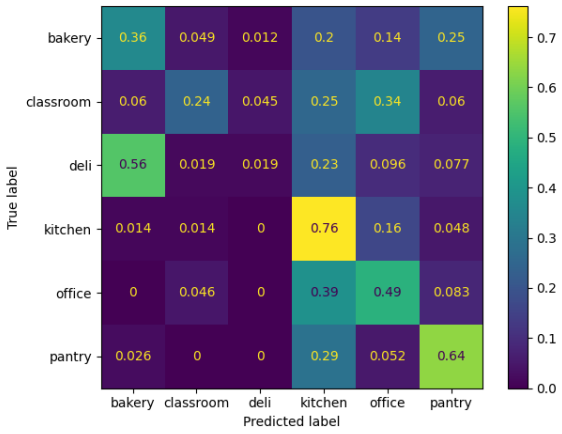

Fig. 7: Confusion matrix from evaluating the ResNet18 network trained on the Indoor dataset on the test split, showing that the model confuses images belonging to the class *deli* with images of *bakery* more than 50% of the time.

for *deli*, where the images feature only baked goods and are more likely to be identified by humans as *bakery* rather than *deli* but were labeled in the dataset as *deli*. This overlap leads to noisy data, complicating the training process and contributing to the misclassification of similar images across different classes.

Proactive bias detection

Our framework can be utilized to identify potential misclassification candidates by using patterns extracted from correctly classified images, denoted as $\mathcal{P}_c^k$. This can be done by leveraging CSKGs to extract antonyms for each node in $\mathcal{P}_c^k$ and then generate all possible antonym pattern combinations. For instance, consider $\mathcal{P}_c^{female} = ((hair, is, long) \wedge (hair, is, blond))$ from CelebA dataset. Based on the CSKG, while *hair* has no antonym, *long* and *blond* have antonyms *short* and *brunette*, respectively. This yields the following antonym patterns: $((hair, is, short) \wedge (hair, is, blond)) \vee ((hair, is, long) \wedge (hair, is, brunette)) \vee ((hair, is, short) \wedge (hair, is, brunette))$.

Images with these antonym patterns are more likely to be misclassified by the image classifier, especially if these patterns are rare or absent in the training data and, therefore, not present in the extracted patterns. This approach was applied to the CelebA and Waterbirds datasets, revealing that the derived antonym patterns align with the filtering used to create unbiased test splits. These test splits often include images that a non-robust classifier misclassifies, as reflected in the low test accuracies in Tab. 2. This underscores the value of using patterns from correctly classified images to identify gaps in the classifier's knowledge.

Additionally, our framework can be used to identify spurious patterns in the dataset before training by setting predicted labels to the ground truth. The extracted patterns can then be manually inspected for biases, serving as a proactive measure to develop fairer and more robust models.

3.4 Model Parameters, Settings and Runtime

This paper employs several models with carefully selected parameters:
• **LLaVA-v1.5-7b**, an open-source multi-modal model, was used without fine-tuning via the Hugging Face implementation, generating up to 300 tokens per output;
• **OpenIE**, an unsupervised extraction tool from the Stanza library (v1.7.0), was configured with affinity probability (1/3) to reduce noise, maximum entailments per clause (2) to ensure relevance, coreference resolution (*True*) for accuracy, and triple all nominals (*True*) to capture nominal relations;
• **CgSpan (v0.2.2)**, applied to mine frequent closed subgraph patterns, used a 50% minimum support threshold and minimum subgraph size of one edge (no maximum), balancing pattern complexity and frequency, with other parameters at default, using the authors' original code.
• The parameter n in our framework is set to $n = 2$. Based on our evaluation (see Tab. 5) increasing n from 1 to 2 substantially enhances both pattern quality and classifier alignment. This choice offers a practical trade-off between granularity and computational efficiency.

All experiments were run on an NVIDIA Tesla V100-SXM2-32GB GPU under Red Hat Enterprise Linux 8.9, providing up to 31.33 TFLOPS (FP16), with runtimes averaging 2 seconds per LLaVA query, circa 100 ms per OpenIE extraction, and under 1 minute per class-level graph processed by cgSpan depending on dataset and settings.

4 Related Work

Systematic error analysis
Recent research has focused extensively on identifying systematic errors in neural networks [38,4,22,31,16,7,12]. Traditional methods use manually defined dimensions, such as gender, ethnicity, or object labels, to identify causes of failure. For instance, [38] relies on object labels to identify correlations between objects and model predictions, while [4] examines biases in facial analysis algorithms based on predefined phenotypes. Other approaches manipulate dimensions like blurriness to evaluate robustness [16] or analyze rare subgroups [31,7]. In contrast, our framework dynamically extracts dimensions and their values in an unsupervised fashion, eliminating manual efforts. It also extracts patterns from these dimensions for correct and incorrect predictions.

While [10] uses image captions to identify commonalities across image groups without fixed dimensions, captions fail to capture detailed structural relationships, which our scene descriptors provide. Captions have also been used to generate failure-inducing images [44,3], but these methods often miss critical scene details. Additionally, clustering misclassified images in latent spaces [9,20,39,46] often leads to incoherent groups, complicating caption associations and corrective evaluations [14].

Scene graphs
The goal of scene graph generation is to parse an image to create a structured representation, ultimately aiming for a complete understanding of visual scenes. The following two approaches are typically used: (1) a two-stage method that first detects objects and

then recognizes relationships [47,5,6,33,50,49,52,32], which gained significant attention with the release of the VG dataset; and (2) a one-stage method that simultaneously detects and recognizes objects and relations [28,8,41,25]. A comprehensive review of scene graph generation methods was conducted in [54]. Existing work focuses primarily on detecting objects and their relationships, often lacking detailed object descriptions (e.g., color, shape) and contextual information (e.g., weather, lighting, image mode, etc.) that could be the cause of model bias. Our unsupervised scene descriptor fills this gap by offering crucial details for a deeper understanding of the visual scene.

Frequent sub-graph mining
Frequent sub-graph mining is a field within data mining dedicated to identifying common sub-graphs across graph datasets. Numerous studies have contributed to this field, [18,48,37,24] (see [21] for an extensive review). We chose cgSpan [37] for our work as it identifies frequent closed sub-graphs, ensuring that there are no supergraphs with the same frequency. This provides a compact, efficient representation of all frequent sub-graphs, reducing redundancy and offering faster processing than methods that enumerate all frequent sub-graphs.

5 Conclusion

We introduced a dynamic, unsupervised framework that generates graph-based patterns, offering deeper and more comprehensive explanations for better understanding of classifier behavior compared to the shallow, manually predefined dimensions used in previous work. The framework comprises two components: a novel scene descriptor that generates detailed and adaptable scene graphs for images, and a pattern extractor that identifies key patterns distinguishing between correct and incorrect classifications. Empirical evaluations demonstrate the robustness and effectiveness of our approach. Our scene graphs are richer and more detailed compared to manual annotations from VG. The pattern miner achieves high F1 scores in predicting classifier behavior, with extracted patterns proving valuable for error analysis and bias detection, fostering fairer and more robust models. This work presents the first approach for generating detailed graph-based explanations for image classifiers in an unsupervised fashion, surpassing prior efforts that rely on simplistic, manually annotated data. In future work we plan to extend our framework to utilize different foundation models and to provide explainability on models pretrained on unknown data.

6 Acknowledgment

This work was partially funded by the European Union's programme under grant agreement No.101092908 (SMARTEDGE).

Supplemental Material Statement: The supplementary material, available at: `https://doi.org/10.6084/m9.figshare.29690348.v2`, includes a link to code and data. TechnicalAppendix.pdf stores additional information regarding the paper and experimental results.

References

1. Angeli, G., Johnson, M., Manning, C.D.: Leveraging linguistic structure for open domain information extraction. In: Annual Meeting of the Association for Computational Linguistics (2015)
2. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NeurIPs. pp. 2787–2795 (2013)
3. Boreiko, V., Hein, M., Metzen, J.H.: Identifying systematic errors in object detectors with the scrod pipeline. ICCVW 2023 pp. 4092–4101 (2023)
4. Buolamwini, J., Gebru, T.: Gender shades: Intersectional accuracy disparities in commercial gender classification. In: Conference on Fairness, Accountability and Transparency, FAT 2018, 23-24 February 2018, New York, NY, USA. Proceedings of Machine Learning Research, vol. 81, pp. 77–91. PMLR (2018)
5. Chen, G., Li, J., Wang, W.: Scene graph generation with role-playing large language models. In: Proceedings of the 38th International Conference on Neural Information Processing Systems (NeurIPS). pp. 4203–4231. Curran Associates Inc., Red Hook, NY, USA (2025)
6. Chen, T., Yu, W., Chen, R., Lin, L.: Knowledge-embedded routing network for scene graph generation. 2019 CVPR pp. 6156–6164 (2019)
7. Collevati, M., Eiter, T., Higuera, N.: Leveraging neurosymbolic AI for slice discovery. In: NeSy 2024. vol. 14979, pp. 403–418 (2024)
8. Cong, Y., Yang, M.Y., Rosenhahn, B.: Reltr: Relation transformer for scene graph generation. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**, 11169–11183 (2022)
9. d'Eon, G., d'Eon, J., Wright, J.R., Leyton-Brown, K.: The spotlight: A general method for discovering systematic errors in deep learning models. ACM FAccT 2022 (2022)
10. Dunlap, L., Zhang, Y., Wang, X., Zhong, R., Darrell, T., Steinhardt, J., Gonzalez, J.E., Yeung-Levy, S.: Describing differences in image sets with natural language. ArXiv **abs/2312.02974** (2023)
11. Efron, B.: Bootstrap methods: Another look at the jackknife. Annals of Statistics **7**, 1–26 (1979)
12. Fontanesi, M., Micheli, A., Podda, M.: Explaining graph classifiers by unsupervised node relevance attribution. In: xAI (2024)
13. Galkin, M., Denis, E.G., Wu, J., Hamilton, W.L.: Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs. In: ICLR 2022 (2022)
14. Gao, I., Ilharco, G., Lundberg, S.M., Ribeiro, M.T.: Adaptive testing of computer vision models. ICCV 2023 pp. 3980–3991 (2023)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CVPR 2016 pp. 770–778 (2016)
16. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. In: 2021 ICCV. pp. 8320–8329. IEEE (2021)
17. Herzig, R., Mendelson, A., Karlinsky, L., Arbelle, A., Feris, R.S., Darrell, T., Globerson, A.: Incorporating structured representations into pretrained vision & language models using scene graphs. ArXiv **abs/2305.06343** (2023)
18. Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: European Conference on Principles of Data Mining and Knowledge Discovery (2000)
19. Ismaeil, Y., Stepanova, D., Tran, T.K., Saranrittichai, P., Domokos, C., Blockeel, H.: Towards neural network interpretability using commonsense knowledge graphs. In: International Workshop on the Semantic Web (2022)

20. Jain, S., Lawrence, H., Moitra, A., Madry, A.: Distilling model failures as directions in latent space. ArXiv **abs/2206.14754** (2022)
21. Jiang, C., Coenen, F., Zito, M.A.A.: A survey of frequent subgraph mining algorithms. The Knowledge Engineering Review **28**, 75 – 105 (2012)
22. Kim, B., Wattenberg, M., Gilmer, J., Cai, C.J., Wexler, J., Viégas, F.B., Sayres, R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018. Proceedings of Machine Learning Research, vol. 80, pp. 2673–2682. PMLR (2018)
23. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., Bernstein, M.S., Fei-Fei, L.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. IJCV **123**, 32 – 73 (2016)
24. Kuramochi, M., Karypis, G.: Finding frequent patterns in a large sparse graph. Data Mining and Knowledge Discovery **11**, 243–271 (2004)
25. Li, R., Zhang, S., He, X.: Sgtr: End-to-end scene graph generation with transformer. CVPR 2022 pp. 19464–19474 (2022)
26. Li, Z., Xu, C.: Discover the unknown biased attribute of an image classifier. 2021 ICCV pp. 14950–14959 (2021)
27. Liu, H., Li, C., Li, Y., Lee, Y.J.: Improved baselines with visual instruction tuning. In: CVPR 2024. pp. 26296–26306 (2024)
28. Liu, H., Yan, N., Mortazavi, M.S., Bhanu, B.: Fully convolutional scene graph generation. CVPR 2021 pp. 11541–11551 (2021)
29. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. ICCV 2015 pp. 3730–3738 (2015)
30. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Softw. **2**, 205 (2017)
31. Metzen, J.H., Hutmacher, R., Hua, N.G., Boreiko, V., Zhang, D.: Identification of systematic errors of image classifiers on rare subgroups. ICCV 2023 pp. 5041–5050 (2023)
32. Mitra, C., Huang, B., Darrell, T., Herzig, R.: Compositional chain-of-thought prompting for large multimodal models. ArXiv **abs/2311.17076** (2023)
33. Qi, M., Li, W., Yang, Z., Wang, Y., Luo, J.: Attentive relational networks for mapping images to scene graphs. 2019 CVPR pp. 3952–3961 (2019)
34. Quattoni, A., Torralba, A.: Recognizing indoor scenes. 2009 CVPR pp. 413–420 (2009)
35. Sagawa, S., Koh, P.W., Hashimoto, T.B., Liang, P.: Distributionally robust neural networks. In: ICLR (2020)
36. Sainz, O., García-Ferrero, I., Agerri, R., de Lacalle, O.L., Rigau, G., Agirre, E.: GoLLIE: Annotation guidelines improve zero-shot information-extraction. In: The Twelfth International Conference on Learning Representations (2024), https://openreview.net/forum?id=Y3wpuxd7u9
37. Shaul, Z., Naaz, S.: cgspan: Closed graph-based substructure pattern mining. In: 2021 IEEE International Conference on Big Data (Big Data). pp. 4989–4998 (2021)
38. Singh, K.K., Mahajan, D., Grauman, K., Lee, Y.J., Feiszli, M., Ghadiyaram, D.: Don't judge an object by its context: Learning to overcome contextual bias. In: 2020 CVPR. pp. 11067–11075. Computer Vision Foundation / IEEE (2020)
39. Sohoni, N.S., Dunnmon, J.A., Angus, G., Gu, A., Ré, C.: No subclass left behind: Fine-grained robustness in coarse-grained classification problems. ArXiv **abs/2011.12945** (2020)
40. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: AAAI Conference on Artificial Intelligence (2016)
41. Teng, Y., Wang, L.: Structured sparse r-cnn for direct scene graph generation. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 19415–19424 (2021)

42. Thrush, T., Jiang, R., Bartolo, M., Singh, A., Williams, A., Kiela, D., Ross, C.: Winoground: Probing vision and language models for visio-linguistic compositionality. 2022 CVPR pp. 5228–5238 (2022)
43. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.P.: Composition-based multi-relational graph convolutional networks. In: ICLR 2020 (2020)
44. Vendrow, J., Jain, S., Engstrom, L., Madry, A.: Dataset interfaces: Diagnosing model failures using controllable counterfactual generation. ArXiv **abs/2302.07865** (2023)
45. Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., Zhou, M.: Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. ArXiv **abs/2002.10957** (2020)
46. Wiles, O., Albuquerque, I., Gowal, S.: Discovering bugs in vision models using off-the-shelf image generation and captioning. ArXiv **abs/2208.08831** (2022)
47. Xu, B., Liao, R., Sigal, L.: Self-supervised relation alignment for scene graph generation. 2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) pp. 1328–1338 (2023)
48. Yan, X., Han, J.: gspan: graph-based substructure pattern mining. 2002 IEEE International Conference on Data Mining, 2002. Proceedings. pp. 721–724 (2002)
49. Yang, X., Liu, Y., Wang, X.: Reformer: The relational transformer for image captioning. Proceedings of the 30th ACM International Conference on Multimedia (2021)
50. Yao, Y., Zhang, A., Han, X., Li, M., Weber, C., Liu, Z., Wermter, S., Sun, M.: Visual distant supervision for scene graph generation. ICCV 2021 pp. 15796–15806 (2021)
51. Yuksekgonul, M., Bianchi, F., Kalluri, P., Jurafsky, D., Zou, J.Y.: When and why vision-language models behave like bags-of-words, and what to do about it? ArXiv **abs/2210.01936** (2022)
52. Zhao, S., Xu, H.: Less is more: Toward zero-shot local scene graph generation via foundation models (2023)
53. Zhao, T., Zhang, T., Zhu, M., Shen, H., Lee, K., Lu, X., Yin, J.: Vl-checklist: Evaluating pre-trained vision-language models with objects, attributes and relations. ArXiv **abs/2207.00221** (2022)
54. Zhu, G., Zhang, L., Jiang, Y., Dang, Y., Hou, H., Shen, P., Feng, M., Zhao, X., Miao, Q., Shah, S.A.A., Bennamoun: Scene graph generation: A comprehensive survey. Neurocomputing **566**, 127052 (2022)