HiKER-SGG: Hierarchical Knowledge Enhanced Robust Scene Graph Generation

Supplementary Material

In this supplementary document, we provide additional details and experimental results to enhance understanding and insights into our proposed HiKER-SGG. This supplementary document is organized as follows:

- The hierarchical clustering process and visualization results in Section 3.2 are shown in Section A.1.
- A more detailed description for both the scene graph and the hierarchical knowledge graph in Section 3.3-3.5 is shown in Section A.2.
- More discussions about the adaptive refinement in Section 3.7 is in Section A.3.
- We provide a more comprehensive description of the experimental settings used in our study in Section B.1.
- A detailed description and visualization of the VG-C dataset are provided in Section B.2.
- We also present more experimental results in Section B.3 and B.4.
- Finally, we discuss the potential limitations of the proposed HiKER-SGG in Section C.

A. More Details about HiKER-SGG

A.1. Hierarchical Clustering

As we introduced in Section 3.2, we use a hierarchical clustering algorithm to discover both the entity and predicate hierarchies. We provide the pseudocode in Algorithm 1. Specifically, hierarchical clustering initializes with individual class names as separate clusters and repeatedly merges the two clusters with the highest similarity until only one cluster remains. During each iteration, it updates the similarity measures of the newly formed cluster with the remaining clusters, ensuring that the most similar clusters are merged at each step. This process results in a hierarchical structure of clusters based on the defined similarity metric.

After completing the hierarchical clustering, we select the three lowest-level clusters to conduct our hierarchical inference process defined in Section 3.6. The discovered predicate and entity hierarchies are visualized in Figure A1. Notably, the discovered hierarchical structure reasonably clusters similar classes in the same superclass, for example, (1) `wearing` and `wears` in predicate classes, and (2) `boy`, `girl`, `child`, and `kid` in entity classes. Although most hierarchical relationships are accurately identified, some may appear noisy from a human perspective. Nevertheless, given that our clustering is based on pre-defined similarity metrics, we do not perform additional cleaning and believe our model is equipped to handle these issues.

In Section 4.3, we have shown that replacing manually configured hierarchical structures with those discovered ones yields a non-trivial 0.4%∼0.7% increase in mR@k metrics. These results demonstrate that the hierarchies uncovered by this method, provide more effective guidance for our hierarchical inference approach.

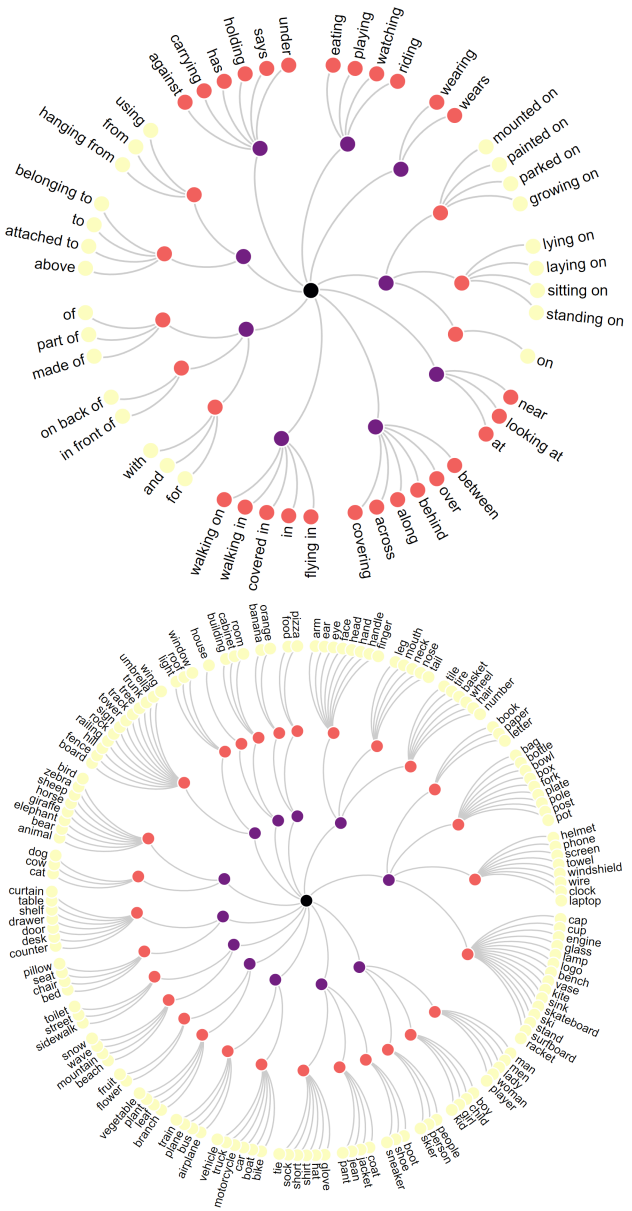

Figure A1. **The visualization of the discovered hierarchies on the Visual Genome [38] dataset**. Top: the discovered hierarchy for 50 predicate classes; Bottom: the discovered hierarchy for 150 entity classes. In this work, we simply utilize 3-level hierarchies for the hierarchical inference process.

Algorithm 1: Hierarchical Clustering Algorithm

Input: Category set $\mathcal{C} = \{\mathcal{C}_i\}_{i=1}^{n}$; Similarity metric $\mathsf{Sim}(\cdot, \cdot)$ defined in Section 3.2.
Output: The hierarchy of clusters L.

Initialize the clusters L with n clusters, each containing a class name;
repeat
  Find pairs of clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ in L with highest similarity $\mathsf{Sim}(\mathcal{C}_1, \mathcal{C}_2)$;
  Merge $\mathcal{C}_1$ and $\mathcal{C}_2$ into a new cluster $\mathcal{C}_{12}$;
  Remove $\mathcal{C}_1$ and $\mathcal{C}_2$ from L;
  for *each cluster* $\tilde{\mathcal{C}} \in L$ **do**
    Update the similarity of the created cluster with other clusters with $\mathsf{Sim}(\mathcal{C}_{12}, \tilde{\mathcal{C}})$;
  end
  Add this new cluster $\mathcal{C}_{12}$ to L;
until $|L| = 1$;

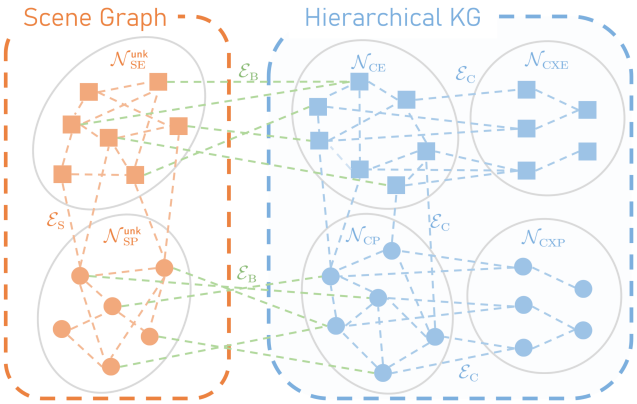

Figure A2. **The architecture and notations of our scene graph and hierarchical knowledge graph**. Nodes and edges within the scene graph are orange, those within the knowledge graph are blue, and the bridge edges that connect the two graphs are green.

A.2. Scene Graph and Hierarchical Knowledge Graph

In Figure A2, we summarize the architecture and notations of our scene graph and hierarchical knowledge graph we construct in this work. Specifically, we have 6 different types of nodes, as well as 3 types of edges. Below, we detail each one individually.

We have 6 different types of nodes:

- Commonsense entity node $\mathcal{N}_{\text{CE}}$ in the knowledge graph. We only consider 150 entities from the VG dataset.
- Commonsense predicate node $\mathcal{N}_{\text{CP}}$ in the knowledge graph. We only consider 50 predicates from the VG dataset.
- Commonsense superclass entity node $\mathcal{N}_{\text{CXE}}$ in the knowledge graph. This includes a set of specialized entity nodes at various levels, corresponding to overarching categories of entities.
- Commonsense superclass predicate node $\mathcal{N}_{\text{CXP}}$ in the knowledge graph. This includes a set of specialized predicate nodes at various levels, corresponding to overarching categories of predicates.
- Scene entity node $\mathcal{N}_{\text{SE}}$ in the scene graph. Derived from the commonsense entity node, each scene entity (SE) node $\mathcal{N}_{\text{SE}}$ is additionally linked with a bounding box, *i.e.*, $\mathcal{N}_{\text{SE}} \subseteq [0, 1]^4 \times \mathcal{N}_{\text{CE}}$.
- Scene predicate node $\mathcal{N}_{\text{SP}}$ in the scene graph. Originating from the commonsense predicate node, each scene predicate (SP) node $\mathcal{N}\text{SP}$ connects a pair of SE nodes, *i.e.*, $\mathcal{N}_{\text{SP}} \subseteq \mathcal{N}_{\text{SE}} \times \mathcal{N}_{\text{SE}} \times \mathcal{N}_{\text{CP}}$.

We also have 3 types of edges to connect these nodes:

- Commonsense edges $\mathcal{E}_{\text{C}}$ in the knowledge graph. These edges within the commonsense graph $\mathcal{E}_{\text{C}}$ delineate relationships between each node pair in both sets, functioning as a reservoir of general knowledge about objects. Examples include connections like `man-wears-shirt` and `cat-is-animal`.
- Scene edges $\mathcal{E}_{\text{S}}$ in the scene graph. These edges encapsulate the relationships within the scene graph, linking scene entities and predicates to denote interactions and spatial relationships in a given scene.
- Bridge edges $\mathcal{E}_{\text{B}}$ connecting commonsense nodes and scene nodes. These bi-directional bridge edges link an entity or predicate from the scene graph to its corresponding labels in the commonsense graph. Given the symmetric nature of the relation, the bridge edges are implemented as bi-directional directed edges with shared weights.

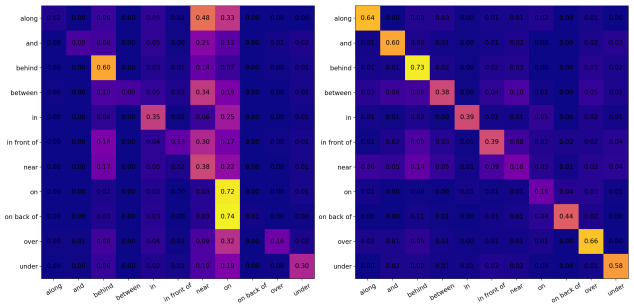

Figure A3. **An illustration of adaptive refinement**. Left: the initial confusion matrix $\mathcal{R}^0$; Right: the confusion matrix $\mathcal{R}^5$ after 5 training epochs.

A.3. Adaptive Refinement

To provide insights into the adaptive refinement process introduced in Section 3.7, we present a visualization of the

initial confusion matrix $\mathcal{R}^0$, along with the updated confusion matrix $\mathcal{R}^5$ after 5 training epochs in Figure A3. The initial confusion matrix $\mathcal{R}^0$ exhibits strong performance in general classes with more samples. In contrast, the updated version $\mathcal{R}^5$ strives for balanced accuracy across all predicate classes. Simultaneously, rather than the initial matrix which aims to reveal surface-level biases, the updated $\mathcal{R}^5$ shifts its focus to uncovering deeper and subtle correlations between predicate classes, as indicated by more minor values off the diagonal (*e.g.*, `near` with `along`/`and`/`between`).

B. More Experimental Results

B.1. Experiment Settings

Tasks. Following previous work [9, 85], we assess the effectiveness of our proposed approach in the context of two standard SGG tasks: Predicate Classification (PredCls) and Scene Graph Classification (SGCls). In the PredCls scenario, our model is provided with ground-truth bounding boxes and their associated object classes, with the sole task of predicting the predicate class. In the SGCls scenario, the model is only provided with known bounding boxes while the object classes are treated as unknown, and our SGG model is required to predict both the object and predicate classes.

Evaluation Metrics. We evaluate the performance of the SGG models by top-k mean triplet recall (mR@k) metric on both the PredCls and SGCls tasks. In specific, mR is the average recall score between the top-k predicted triplets and ground-truth ones across all 50 predicate categories, which promotes unbiased prediction for less frequently occurring predicate classes. A subject-predicate-object triplet is considered a match when all three components are correctly classified, and the subject and object bounding boxes align with an IoU (Intersection over Union) score of at least 0.5. In our experiments, we report the mean recall on $k = 20, 50, 100$ to comprehensively evaluate the effectiveness of our method. We also report the constrained (C) and unconstrained (UC) performance results, depending on the presence or absence of the graph constraint. This constraint restricts our SGG model to predict only a single relation between each pair of objects.

Implementation Details. We use the Faster-RCNN [56] as the object detector, which is based on VGG-16 [58] backbone provided by Zellers *et al.* [87]. In our experiments, we train our model for 30 epochs, initializing the learning rate at 1×10^{-4}. This learning rate will decrease to $1/10$ of its value after every 10 epochs. A single NVIDIA Quadro RTX 6000 GPU is used for all the experiments.

Fairness. To the best of our knowledge, our work is the first to tackle the robustness challenge in SGG, therefore there are no other established baselines for this task available. However, we do our best to ensure a fair comparison: all models rigorously follow the ***same*** evaluation protocol stated in Section 4. Our experiments are designed to highlight: (1) Compared to GB/EB-Net, HiKER-SGG enables a more comprehensive and efficient exploitation of KG information. (2) Compared to other methods, the performance gain demonstrates the effectiveness of KG in enhancing SGG. To ensure a fair comparison with non-graph-based methods, we also conduct an experiment that set the message propagation steps as $t = 0$ to isolate the effect of KG. In the PredCls tasks, the mR@50/100 accuracy remains competitive at 34.9%/37.1%.

Corruption Type	Abbreviation
Gaussian Noise	gaus
Shot Noise	shot
Impulse Noise	imp
Defocus Blur	dfcs
Glass Blur	gls
Motion Blur	mtn
Zoom Blur	zm
Snow	snw
Frost	frst
Fog	fg
Brightness	brt
Contrast	cnt
Elastic	els
Pixelate	px
JPEG Compression	jpg
Sunlight glare	sun
Water drop	wtd
Wildfire Smoke	smk
Rain	rain
Dust	dust

Table B1. **Abbreviations** of the 20 corruption types in our created corrupted Visual Genome (VG-C) benchmark.

B.2. Corrupted Visual Genome Benchmark

In addition to the clean Visual Genome dataset, we also evaluate our method on the corrupted Visual Genome [38] (VG-C) dataset, which comprises 20 versions of corrupted images designed to simulate realistic corruptions that may occur in real-world scenarios, thereby providing insights into the models' robustness under various corruption conditions. Of these corruptions, the first 15 types of corruption introduced by Hendrycks *et al.* [28] are widely recognized as standard benchmarks for evaluating robustness within the research community. To further align with real-world deployment scenarios, we introduce 5 additional types of *natural* corruptions to our evaluation:

- **Sunlight glare**: Sunlight glare refers to the interference caused by excessive sunlight or bright light sources in an image. It typically results in overexposed or washed-out areas in the photo, making it difficult to discern details and colors.

Table B2. **Multi-hop accuracy on the PredCls task using the Visual Genome [38] dataset**. We compare our method with EB-Net [9] method, assessing performance based on both level-1/2 superclass and final subclass accuracy.

	Setting	mR@20: UC/C	mR@50: UC/C	mR@100: UC/C
EB-Net	1-hop	51.6 / 50.5	68.2 / 63.7	79.4 / 68.1
	2-hop	45.4 / 40.2	62.8 / 48.9	73.7 / 52.0
	3-hop	39.8 / 30.8	54.9 / 36.7	66.3 / 39.2
Ours	1-hop	59.6 / 57.8	75.6 / 69.1	87.7 / 75.3
	2-hop	50.8 / 45.2	67.7 / 53.8	79.6 / 57.2
	3-hop	42.1 / 33.4	57.9 / 39.3	69.2 / 41.2

- **Water drop**: Water drop corruption occurs when water droplets or condensation obstruct the camera lens or affect the image sensor. This can create blurry or distorted portions of the image and often results in a hazy or unfocused appearance.
- **Wildfire smoke**: Wildfire smoke corruption pertains to images taken in areas affected by heavy smoke. It causes reduced visibility, a haze or smoky appearance, and can obscure objects in the frame.
- **Rain**: Rain refers to the presence of falling raindrops in an image. Rain can cause blurriness and distortions, making it difficult to see objects clearly.
- **Dust**: Dust corruption results from particles or dust settling on the camera lens or sensor. This can lead to the appearance of dark spots or specks in the image, which may obscure details and reduce clarity.

We establish 5 distinct severity levels for each corruption, following Hendrycks *et al.* [28] to facilitate future benchmarking. Table B1 presents a summary of the abbreviations used for the various types of corruption. To illustrate the effects of these corruptions, we present the corrupted versions of two example images in Figure B4.

We have already made the processing code for generating these corruptions available (VG-C benchmark) at https://github.com/zhangce01/HiKER-SGG. This benchmark offers a comprehensive evaluation platform to assess the robustness of SGG models in adverse conditions, and we encourage the formulation of new SGG models to be evaluated using this benchmark, emphasizing the real-world applications of the SGG task.

B.3. Multi-Hop Accuracy

To further illustrate the robustness of our method, we compare HiKER-SGG with EB-Net [9] by multi-hop mean recall metrics on the Visual Genome dataset in Table B2. Our evaluation criterion is as follows: a 1/2-hop prediction is considered correct if any of the final predicted predicate classes in the triplets correspond to the true level-1/2 superclass. By designing our model to predict from higher to lower levels, our HiKER-SGG not only achieves state-of-the-art performance in final subclass prediction, but also exhibits superior performance in 1/2-hop superclass prediction, outperforming the baseline method by an average of 8% and 5% in mean recall, respectively. This performance highlights that when unable to classify to the final subclass, HiKER-SGG tends to more accurately predict the superclass, illustrating the robustness of our hierarchical prediction approach.

Table B3. Training time, testing time, and parameter count of HiKER-SGG compared with other methods.

Method	Training (/epoch)	Inference (/image)	# params
KERN [8]	179.1 min	0.32 s	405.2M
GB-Net [85]	84.6 min	0.20 s	444.6M
EB-Net [9]	89.7 min	0.22 s	448.8M
HiKER-SGG	101.3 min	0.24 s	455.9M

B.4. Inference Time

In Section 4.3, we have shown that our HiKER-SGG exhibits significantly enhanced robustness with both clean and corrupted images with only about 10% training costs. In Table B3, we also include inference time form comparisons. A single NVIDIA Quadro RTX 6000 GPU is used for all the experiments. When compared to state-of-the-art methods such as GB-Net [85] and EB-Net [9], the HiKER-SGG model only extends the inference time by a slight 0.02-0.04 seconds. This minor increase is likely negligible in practical real-world deployment scenarios.

C. Limitations

We identify two potential limitations of our HiKER-SGG method: (1) For each new dataset, a hierarchical structure must be re-discovered, potentially increasing complexity. Additionally, the selection of similarity metrics also includes bias or the prior incorporation by humans. We acknowledge that the choice of measures does reflect a one-time prior human incorporation. However, once determined, the process becomes systematic. This is fundamentally different from the continuous, subjective interventions that characterize the human bias we aim to avoid. (2) Our method is tested in corrupted experiments on PredCls and SGCls tasks, assuming the accuracy of detected bounding boxes. However, in cases of severely corrupted images where the object detector fails to recognize objects, our HiKER-SGG method may not perform effectively. However, in our experiments, a simple Faster-RCNN is able to identify nearly 50% of the GT boxes even under corrupted scenarios; In contrast, given the GT boxes, SGG models can only achieve about 11% mR@100 in SGCls task. This highlights the practical significance of enhancing the robustness of SGG models. Besides, we also notice that there is another line of work and benchmarks (*e.g.*, Foggy Cityscapes) focusing on designing robust detectors. Combining our approach with them could further enhance the overall reliability of the system.

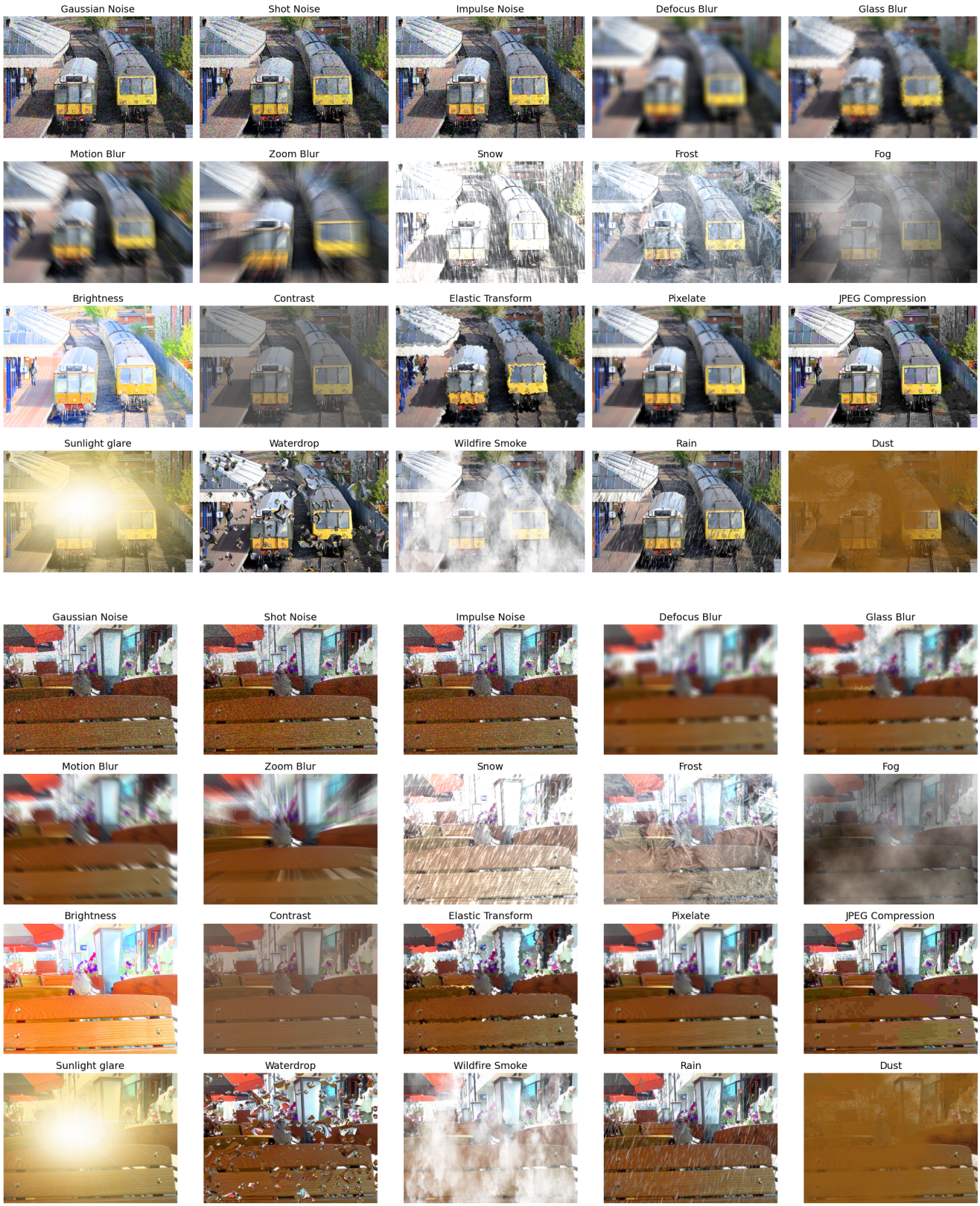

Figure B4. **All the 20 corruption types we used in our corrupted experiments**. The first 15 types of corruption are introduced by Hendrycks *et al.* [28], and we introduce 5 additional types of *natural* corruptions for a more comprehensive and practical evaluation.