Learning Visual Commonsense for Robust Scene Graph Generation: Supplementary Material

Alireza Zareian*, Zhecan Wang*, Haoxuan You*, and Shih-Fu Chang

Columbia University, New York NY 10027, USA
{az2407,zw2627,hy2612,sc250}@columbia.edu

In the following, we provide additional details and analysis that was not included in the main paper due to limited space. We first provide more implementation details, followed by a novel evaluation protocol that reveals insightful statistics about the data and the state of the art performance. We finally provide more qualitative examples to showcase what our commonsense model learns.

1 Implementation Details

We have three training stages: perception, commonsense, and joint fine-tuning. We train the perception model by closely following the implementation details of each method we adopt [2, 3, 1], with the exception that for IMP [2], we use the implementation by Zellers *et al.* [3], because it performs much better. We separately train the commonsense model (GLAT) once, independent of the perception model, as we described in the main paper. Then we stack GLAT on top of each perception model and perform fine-tuning, without the fusion module. Finally, we add the fusion model for inference.

Our GLAT implementation has 6 layers each with 8 attention heads, each with 300-D representations. We train it with a 30% masking rate on Visual Genome (VG) training scene graphs, using an Adam optimizer with a learning rate of 0.0001, for 100 epochs. To stack the trained GLAT model on top of a perception model, we take the scene graph output of the perception model for a given image, keep the top 100 most confident triplets and remove the rest, and represent each remaining entity and predicate with a one-hot vector that specifies the top-1 predicted class. We intentionally discard the class distribution predicted by the perception model, to let the commonsense model reason independently in an abstract, symbolic space. Perception confidence is later taken into account by our fusion module.

The resulting one-hot graph is represented in the same way as a VG graph, that we have pretrained GLAT on, and is fed into GLAT without masking any node. The GLAT decoder predicts new classes for each node, and new edges, but we ignore the new edges and keep the structure fixed. Hence, the output of GLAT looks like the output of the perception model with the exception that

* Equal contribution.

Table 1. Performance comparison in various levels of triplet frequency, in terms of R@100 (%) for SGCls. # and % stand for absolute and relative frequency respectively.

Statistic/Method	Frequency Bins						Average
# instances in train data	1-3	4-9	10-27	28-81	82-243	243-369	-
# unique triplets in bin	86247	21994	4937	766	89	4	-
% unique triplets in bin	75.6	19.3	4.3	0.7	0.1	0.00004	-
Total % of test data	14.7	21.0	28.0	22.5	10.6	3.1	-
IMP [2]	13.2	23.0	34.8	45.7	58.2	78.2	36.1
IMP + GLAT	13.2	23.0	34.8	45.9	58.5	78.7	37.0
IMP + GLAT + Fusion	**13.3**	**23.1**	**35.0**	**46.2**	**58.7**	**79.2**	**37.4**
SNM [3]	15.0	24.8	36.6	48.4	58.4	74.9	37.8
SNM + GLAT	15.1	24.8	36.6	49.0	58.4	75.2	37.9
SNM + GLAT + Fusion	**15.1**	**24.8**	**36.7**	**49.5**	**58.5**	**75.3**	**38.0**
KERN [1]	16.7	26.2	37.5	48.4	59.6	77.1	38.8
KERN + GLAT	16.7	26.6	37.5	48.4	59.6	77.6	38.8
KERN + GLAT + Fusion	**16.7**	**26.8**	**37.5**	**48.5**	**59.7**	**78.1**	**38.8**

the classification logits of each node are changed. We perform 25 epochs of fine-tuning with 0.00001 learning rate and Adam, using the same entity and predicate loss that is typically used to train SGG models [1].

2 Quantitative Evaluation

Here we revisit the conventional evaluation process in the SGG literature, analyze its limitations, and provide an alternative to use in future work. Xu *et al.* [2] originally used overall recall (R@50 and R@100), which means for each image, they get the top 50 (or 100) triplets predicted by their model, compare to the ground truth triplets of that image, compute recall (number of matched triplets divided by the number of ground truth triplets), and average over all images. Later Chen *et al.* [1] revealed that since ground truth triplets in Visual Genome (both in train and test splits) have highly disproportional statistics, overall recall does not necessarily measure the usefulness of the model. In fact, a simple heuristic that always predicts the most frequent relationship for each pair of object classes, based on a fixed lookup table computed over training data (frequency baseline in [3]), performs not much worse than the state of the art of that time, MotifNet [3].

Chen *et al.* [1] proposed an alternative metric, mean recall (mR@50 and mR@100), in which ground truth triplets are divided into 50 bins, based on their predicate type, the recall is computed for each bin separately, averaged over images, and then averaged over the 50 bins. This way, frequent predicates do not dominate the performance, and simplistic models are not praised for merely picking up bias.

Nevertheless, mean recall does not completely solve the imbalance problem, since even within each bin (predicate type), some compositions are much more common than others. For instance, since VG has a focus on sports, the triplet `person holding racket` dominates the bin `holding`, while `person holding cellphone` has much lower frequency, although intuitively more common in real world. Instead of dividing triplets based on their predicate type, we propose to divide them based on the frequency of each triplet in training data, which highly correlates with the frequency in test data as well. This way, each bin consists of triplets with roughly the same frequency, and no triplet can dominate others. After computing recall for each bin, we also report the average over bins, which can be seen as a triplet-balanced version of mean recall. Using trial and error, we found the best strategy is to divide bins in logarithmic scale, using powers of 3. This way we will not have too few or not too many bins.

Table 1 shows the statistics of each bin in our proposed evaluation setting. Despite the logarithmic scale, we still observe a significant imbalance in the dataset. Specifically, our first bin consists of the rarest triplets, which only appear between 1 and 3 times in training data, and comprise 14.7% of all triplets in the test set of VG, 75.6% if we count each unique triplet only once. The state of the art [1] only achieves 16.7% recall on that significant portion of data, while achieves 77.1% recall on the last bin (4 most frequent triplets, 3.1% of the test set and 0.00004% of unique triplets). This strong disproportion suggests how conventional methods over-invest on few unimportant triplets, at the expense of a large portion of (rare but plausible) real-world situations. Accordingly, we believe our new evaluation metric would encourage further research aiming to close the gap.

3 Qualitative Results

To provide more diverse cases of commonsense learned by our model, Figure 1 shows additional examples of corrections made by our GLAT model to perceived scene graphs. In example (a), the perception model mistakes the `giraffe`'s `head` for a `dog`, but our commonsense model corrects that since `head of giraffe` makes more sense than `dog of giraffe`. The fusion module correctly prefers the output of the commonsense model, due to its higher confidence. In (b), the perception model mistakes the `train` for an `engine`, possibly due to the abnormal color palette. Our commonsense model corrects that since `wheel` more likely belongs to a `train`. In (c), the `vase` has an unusual shape, and is mistaken for a `glass` by the perception model, also because it is made of `glass`, but the commonsense model takes into account the fact that the "`glass`" is `holding` a `flower`, which is what `vase`s do. Moreover, in (d), the `pizza` is visually `on` the cardboard, but technically `in` the box. We, humans, know the latter based on our past experience, and so does the proposed commonsense model.

In example (e), the `man` is `holding` the `laptop` and `using` it at the same time. The perception model predicts `holding` because it is a more visual concept, while the commonsense model predicts `using` which is more a abstract concept, and in fact a more salient and important verb here. In (f), the image is not very clear, and there is no visual distinction between the `boy`'s body

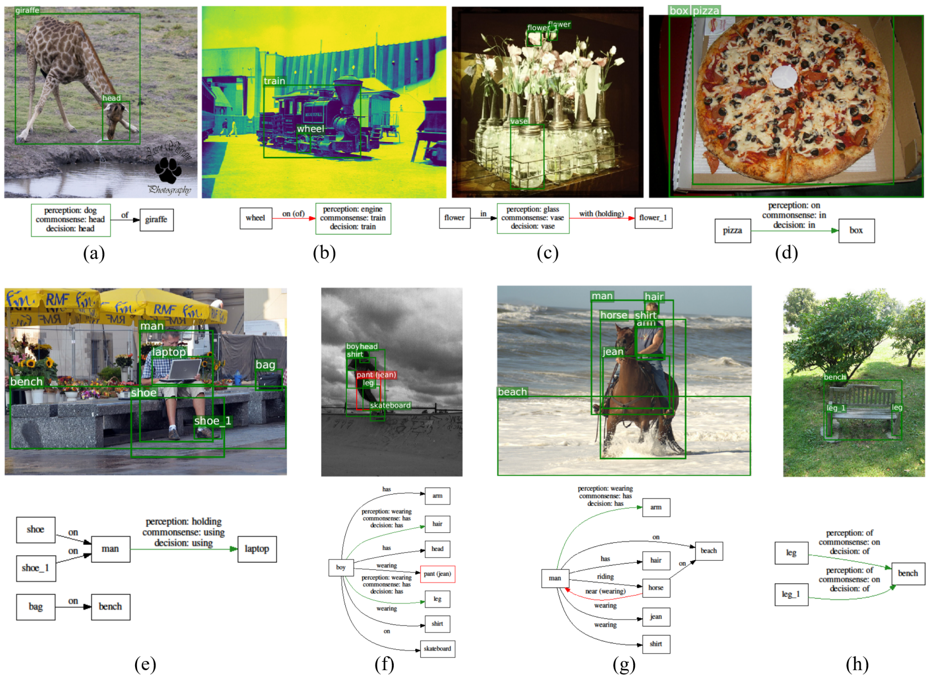

Fig. 1. Example scene graphs generated by the perception, commonsense, and fusion modules, merged into one graph. Entities are shown as rectangular nodes and predicates are shown as directed edges from subject to object. For entities and predicates that are identically classified by the perception and commonsense model, we simply show the predicted label. But in cases where the perception and commonsense models disagree, we show both of their predictions as well as the final output chosen by the fusion module. We show mistakes in red, with the ground truth in parentheses.

parts and clothing. This makes it hard for the perception model to tell the `boy` is `wearing` the `pants` but `has` the `leg`. The commonsense model is robust in such scenarios because it does not rely on the image, but instead considers past experience. Similarly, (g) makes it hard to distinguish `man has arm` from `man wearing shirt`, since the bounding box of `arm` is highly overlapping with `shirt`, and there is little visual distinction between their content. Hence, commonsense has a crucial role in distinguishing their interactions by abstracting them into symbolic concepts and ignoring their visual features. Finally, (h) is a rare case, where the prediction of the commonsense model is wrong, while perception's output was already correct. More specifically, we *miscorrect* `leg of bench` to `leg on bench`, because usually things are `on bench`es in real world. This is while for perception, it is obvious that the `leg`s are not positioned `on` the `bench`. Interestingly, Our fusion module prefers the perceived output this time, and rejects the change made by the commonsense model.

References

1. Chen, T., Yu, W., Chen, R., Lin, L.: Knowledge-embedded routing network for scene graph generation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6163–6171 (2019)
2. Xu, D., Zhu, Y., Choy, C.B., Fei-Fei, L.: Scene graph generation by iterative message passing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5410–5419 (2017)
3. Zellers, R., Yatskar, M., Thomson, S., Choi, Y.: Neural motifs: Scene graph parsing with global context. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5831–5840 (2018)