Thank you to the reviewers for their detailed comments. We commit to addressing the minor and typographical errors. In the following, we discuss the main issues raised:

1. (R2) "if the d-DNNF is already compiled there must be a justification for why the logical embedding is needed" and **"compare a more direct approach... like semantic loss (Xu et al 2018), and use the d-DNNF directly in the loss to enforce constraints without embedding the formula."**

Re: As suggested, we compared our method against using d-DNNFs directly via the semantic loss proposed in (Xu et al. 2018)[1] on the visual relation prediction (VRP) task. Our method outperforms the semantic loss (Table 1).

Table 1: Comparison with semantic loss on visual relation prediction (VRP) task.

Model	Top-5 Acc.(%)
w/o semantic loss and w/o our embedder	84.30%
with semantic loss	84.76%
with GCN embeddings (ours)	$\mathbf{92.77}\%$

A potential reason for the performance difference is that the constraints in VRP are more complicated than those explored in (Xu et al., 2018): there are thousands of propositions and a direct use of d-DNNFs causes the semantic loss to rapidly approach ∞. In contrast, our embedding approach avoids this issue. Moreover, our approach enables generalization to constraints that involve previously unseen propositions; we can leverage representations such as GLoVE word embeddings, which is not possible in the semantic loss. We will add these results and discussion to the paper.

2. (R2) "Building d-DNNFs is a difficult problem ... this difficulty is completely hidden in the paper"

Re: Indeed, building d-DNNFs is a difficult problem in general and we will make this fact clearer in the paper. Practical compilation has progressed significantly thanks to research in the area (e.g., the prior work pointed out by R2). We use `c2d` (Darwiche, 2004) that can compile relatively large d-DNNFs in reasonable time; in our experiments, it took less than 2 seconds to compile a d-DNNF from a CNF with 1000 clauses and 1000 propositions on a standard workstation.

We would like to emphasize that our GCN can embed other logic forms expressible as graphs. For cases where d-DNNF compilation is not possible or prohibitive, other logic forms (e.g., CNF) could be used. The key contribution of our approach is improved accuracy by embedding logical constraints. As suggested by the reviewer, we will highlight the trade-off between accuracy and cost (to obtain compiled forms) in the final version.

3. (R3) "Why not combine them jointly so that ... the framework becomes end-to-end?"

Our framework can be trained end-to-end. However, we train the networks separately to (i) alleviate potential loss fluctuations in joint optimization which makes training easier, and (ii) enable the same logical embeddings to be used with different target networks (for different tasks). As suggested by the reviewer, the networks could be trained end-to-end to fine-tune the embeddings further for a specific task.

4. (R1&R3) "... more discussion of why the low and high cases don't show as much improvement as the moderate case"

Re: We will provide additional discussion in the revision. In brief, we attempted to provide a good coverage of potential problem difficulty. The "low" case represents easy problems that all the compared methods were expected to do well on, and the "high" case represents very challenging problems that all methods were expected to struggle on. We posit that the difference on the low and high regimes were smaller because (i) in the low case, all the methods performed reasonably well, and (ii) on the high regime, embedding the constraints helps to a limited extent and points to avenues for future work.

5. (R3) "Comparison with some state-of-the-art neuro-symbolic methods, e.g. Logic Tensor Networks, could be a plus"

Re: Neuro-symbolic methods, such as Logic Tensor Network (LTN) and logic circuits, are not directly applicable in our setting since the constraints for each input may differ. For example, in the visual relation task, the constraints for each image was different. One possible approach would be to conjunct all the constraints, but there would be 7×10^5 propositions and $\approx 10^5$ clauses, which results in a prohibitively large logic network.

6. (R3) "Provide more details on the methodology such that it could be better understood and enhance its reproducibility." and **"explain how they chose some of their parameters."**

Re: We used grid search to find reasonable parameters; we will state this and provide additional methodological details. We have also submitted our implementation code which can be used to reproduce all the results in the paper.

[1] We used publicly available code provided by the authors.