We thank the reviewers for their comments, and will incorporate their suggestions to improve the paper.

Response to Reviewer 1: Regarding task-specific vs general-purpose embeddings: there is an important role for both kinds of techniques. General-purpose embeddings may be valuable for discovering properties of the graph. However, state of the art performance for any given problem almost always requires fine-tuning the embeddings for a particular task (e.g., recent results for link prediction [6, 7] or semi-supervised classification [4, 2]). Our contribution is a way of achieving effective end-to-end embeddings for substantially more complex problems involving discrete optimization.

Regarding whether an embedding layer is necessary: to our knowledge, all modern learning-based systems for discrete optimization tasks (e.g., [3, 1, 5]) first embed any discrete inputs into a continuous domain in order to harness the power of deep networks and gradient-based training. In some cases, this embedding has been trivial because only Euclidean graphs were considered [1, 5], while others used explicit embedding layers to handle arbitrary graph structures [3]. The motivation for our architecture is to allow embeddings for arbitrary graphs to be customized to combined learning + optimization tasks.

Response to Reviewer 2: Regarding the range of problems we consider: we remark that the set of problems in the paper already span a wide range of algorithm design paradigms: we compare to spectral methods, greedy maximization, recursive partitioning schemes, etc. The overarching problem classes that our method applies to also span a wide range of applications: community detection, maxcut, facility location, influence maximization, and immunization problems, just to name a few examples.

However, our method is not limited just to the problem classes considered in the paper; any problems where the clustering layer's output can be interpreted as a soft solution is eligible. Exploring additional applications of our method would be an interesting topic.

Regarding closed-form loss functions: we selected this loss function, based on independently rounding each coordinate, specifically because it often results in a closed-form loss (i.e., this is a deliberate advantage to our approach). However, in cases where the loss is not available in closed form, it is always possible to draw samples from the independent rounding scheme and apply the REINFORCE estimator. Each sample only requires evaluating the objective function, which is typically cheap relative to backpropagation.

Response to Reviewer 3: Thanks for your comments! If accepted, we would use part of the additional page to add a conclusion.

References

[1] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

[2] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In *NIPS*, 2017.

[3] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song. Learning combinatorial optimization algorithms over graphs. In *NIPS*, 2017.

[4] T. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[5] Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *ICLR*, 2019.

[6] M. Schlichtkrull, T. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, 2018.

[7] M. Zhang and Y. Chen. Link prediction based on graph neural networks. In *NIPS*, 2018.