We thank the reviewers for their valuable comments. All three reviews appreciated the significant reduction in wall-clock time compared with Gurobi in solving several benchmark ILP problems.

General Comment on Novelty. We want to first globally address the concern on novelty in the paper. Identifying novel applications & capabilities of machine learning has long been valued at top machine learning venues. Recent examples include: [Wei et al., 2020] that optimizes proximal solvers using standard RL approaches (and won outstanding paper at ICML 2020); [Balunovic et al., 2018] that optimizes SMT solvers using DAgger; and [Gasse et al., 2019] learns branching decisions in ILP solvers using behavior cloning and graph NNs. In terms of empirical significance, our work is the first to significantly outperform a state-of-the-art commercial solver such as Gurobi in wall-clock for general ILP problems, and we do so by identifying the large neighborhood search framework as a suitable one for incorporating learning. As a result, we believe our work has identified a novel application, learning decompositions for large neighborhood search (LNS), and obtained convincing empirical results to be highly relevant to the NeurIPS community. Next, we address individual comments from each reviewer.

R2. Computational costs. At test time, all the experiments were carried out on the same hardware with 16 logical-core Intel(R) Xeon(R) CPU E5-2637 v4 @ 3.50GHz processor and 132 GB of RAM. At training time, RL costs more to train than IL, while random does not need training. One is normally open to spending training time to obtain improved test time performance, as shown by improvements of IL over random.

R3. Scalability. This is an important question to answer for practical relevance. However, the number of integer variables and constraints is only a rough measure on how hard an ILP instance is [Van Roy and Wolsey, 1987]. Within the benchmark problems we considered, the sizes are already difficult as indicated by long running time of Gurobi. Furthermore, the considered sizes are comparable and, in some cases, exceeding those in recent learning-augmented ILP solver papers [Gasse et al., 2019, He et al., 2014]. A related scalability issue concerns how feasible it is to learn decompositions of millions of variables into thousands of subsets. We believe our current method has limitations at such scales, and studying extensions of our approach (e.g., hierarchical imitation learning) is an interesting future direction.

R3. Choice of k. For larger k, each sub-problem becomes easier to solve, at the expense of smaller neighborhoods, thus reducing the opportunities to find better solutions per iteration. So it is a trade-off of finding out the largest sub-problem that is still amenable to ILP solvers while allowing for the maximal neighborhood space for solution improvement.

R4. Novelty. Please see the general comment above. Our strong empirical results showcase the value of large neighborhood search as a framework/application for incorporating learning.

R4. Choice of RL algorithm. In retrospect, the inclusion of REINFORCE did not convey much information as our emphasis was on imitation learning approaches since they were more effective. This discovery is consistent with other recent works on speeding up ILP solvers [He et al., 2014, Gasse et al., 2019] which employed imitation learning. We included REINFORCE for completeness and will improve the writing to focus more on imitation learning.

R4. SCIP. We thank the reviewer for the suggestion. We ran some experiments on using SCIP as the base solver for the same combinatorial auction instances from regions on 2000 items and 4000 bids. The results are consistent with those using Gurobi: LNS methods outperform SCIP and learning delivers further improvements. The reason we focused on Gurobi in the paper is because it is by far the fastest ILP solver and we were excited by convincingly outperforming it with a general framework. We hope the results on SCIP can convince the reviewer that our method is indeed solver agnostic. We are happy to include a full suite of experiments on SCIP in the final version of the paper.

SCIP	-86578.38 ± 606.21
Random-LNS	-98944.90 ± 645.23
BC-LNS	-100513.84 ± 702.05
FT-LNS	-100913.77 ± 681.00

R4. Addtional feedback. Solve sub-problem: we use a solver, e.g., Gurobi or SCIP, to solve the sub-problem, which is an ILP as well. **REINFOCE samples:** you are correct – it is very computationally expensive, which is another reason we decided to focus on imitation learning. **Tuning parameters:** we used 50 training instances.

References

M. Balunovic, P. Bielik, and M. Vechev. Learning to solve smt formulas. In *NeurIPS*, 2018.

M. Gasse, D. Chételat, N. Ferroni, L. Charlin, and A. Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *NeurIPS*, 2019.

H. He, H. Daume III, and J. M. Eisner. Learning to search in branch and bound algorithms. In *NeurIPS*, 2014.

T. J. Van Roy and L. A. Wolsey. Solving mixed integer programming problems using automatic reformulation. *Operations Research*, 35(1):45–57, 1987.

K. Wei, A. Aviles-Rivero, J. Liang, Y. Fu, C.-B. Schnlieb, and H. Huang. Tuning-free plug-and-play proximal algorithm for inverse imaging problems. In *ICML*, 2020.