We thank reviewers for their time and valuable comments. We will try to answer the concerns raised by each reviewer.

We thank R1 for their positive feedback. F3 and T2 are indeed results on *small* instances. We will clarify the captions.

We thank R2 for their valuable comments. **1)** We respectfully disagree on 1., as we do not consider this as a weakness of the paper. As we explained in l.52-60, state-of-the-art MILP solvers are CPU-based, and performance of these solvers is benchmarked in terms of solving time. As such, fast inference is crucial for any ML-based branching strategy to be competitive w.r.t. this metric. Devising an effective model that can learn how to branch accurately is one thing, and making such a model practical, in realistic scenarios where practitioners do not have access to high-end GPU cards, is another. The high inference cost of GNNs was one of the main limitations for integration within MILP solvers, which we believe we addressed in the submission. Our proposed hybrid model offers a better computational trade-off, while retaining most of the accuracy and generalization ability of GNNs. We propose to make a better case of this question, which we will discuss more thoroughly in a new sub-section "The accuracy/complexity trade-off". **2)** We also respectfully disagree on 2. Taken individually, it is true that each part of our proposed methodology has been proposed in a different ML context. However the core idea of running an expressive, but expensive, model once at the root node, and using the resulting transformation to modulate a simpler model everywhere else in the tree, is novel. We believe it is a natural idea, specific to the problem of deploying ML models in time-sensitive B&B environments, which the ML/OR community will find interesting. The paper shows that this intuitive idea is practical, and promising for the field. **3)** We agree on 3., as our training protocol improvements have only a minor impact on the final performance of the model for branching. We conducted additional preliminary experiments (Table A1), and it seems like KD has an effect on generalization which is visible only on hard instances, while AT has a globally positive effect (except for one scenario in Table A1). We propose to lower our claim from "substantial" to "minor", and we will conduct and report the complete experiment in the final version of the paper, including the small instances and the other 2 problem families. However, we would like to point out that the training protocol improvements do not constitute the main contribution of the paper. **4)** Finally, we thank R2 for the suggestion regarding l.255, which more accurately reflects the behaviour of hybrid-SVM models.

Table A1: Effect of training protocols on B&B tree size for the FiLM model (geometric mean over commonly solved instances).

	cauctions		facilities	
	medium	big	medium	big
e2e	702	8939	340	339
e2e & KD	720	8846	343	**325**
e2e & KD & AT	**686**	**8711**	**336**	345

We thank R3 for their constructive comments. **1)** We agree that our training protocol tweaks do have a minor impact, and we will make this more clear in the text (see response 3 to R2). **2)** Regarding the trade-off between model accuracy and complexity, we agree that our discussion in l.52-60 is too brief, and will put more emphasis on this discussion in a new sub-section (see answer 1 to R2). **3)** Regarding KD, we will report additional experiments to better highlight its contribution (see answer 3 to R2). **4)** We agree that a separate experiment with only AT would bring value to the paper. We will conduct it and report the results in the supplementary materials, or the main paper if space permits. **5)** We will fix the two abstract misphrasings, thank you. **6)** We did measure the suggested additional evaluation metric, optimality gap at time out (see Table A2). The results are again in favor of our proposed approach, we will include them in the paper. Thank you for the suggestion.

Table A2: Mean optimality gap (lower the better) of commonly unsolved "big" instances (number of such instances brackets).

	setcover (33)	indset (39)
FSB	0.170944	0.075543
PB	0.071286	0.029791
RPB	0.062841	0.025186
COMP	0.074041	0.025214
GNN	0.103906	0.034104
FiLM	**0.059692**	**0.018685**

We thank R4 for their insightful comments. **1)** The proposed method is indeed specifically trained on one problem type, which is a very valid point that we do not discuss in the paper. We follow the same setting as Gasse et al., who consider this scenario as relevant because most practitioners usually only care about solving very specific problem types. We will add a reference to that argument to clarify this point. **2)** Regarding "different problem types require different models", this might be true, but here we provide evidence that goes in the other direction. Indeed, the same model seems to work reasonably well on a variety of problem types, without the need for specific adaptations (except for re-training). **3)** We agree that our model does not yield a general MILP branching strategy. Attaining this goal with ML would be a substantial achievement, and indeed is not an easy task. We will include this remark in the concluding section. **4)** Our comparison to general MILP branching strategies like RPB is not completely fair. It is a very good point. However, developing specialized versions of such strategies is a challenge in itself, and would require a dedicated paper. We would gladly compare to such approaches, if available. Thank you for raising this point, and we will include this discussion as well. **5)** Regarding online learning, we do not share the same view here. Those branching rules exploit non-local LP information to improve future decisions based on past branching decisions. But our model exploits such information as well, as pseudo-cost is also used as an input feature, at every node. Online learning would imply changing the branching strategy behaviour given the exact same features, which, in our knowledge, none of the branching strategies do.

Finally, both R3 and R4 do not consider our work reproducible, despite the source code being provided. We take note, and will provide more exhaustive implementation details in the appendix of the final paper.