Summary: We genuinely appreciate all three reviewers' (#2, #4, #5) valuable suggestions to strengthen our paper. Below please find the responses to the specific comments.

▷ **Reviewer #2. Q3: Evaluation Metric.** *Reply:* We apologize for term confusions. We actually used the same metric, micro F1 score, for all methods. When reporting F1 scores, we used all SOTAs' released codes and published settings. The differences between reproduced and original results were previously observed for GraphSAGE and FastGCN too, due to well-known GCN training instability. For instance, GraphSAGE F1 scores were 94.6% (Reddit) according to the FastGCN paper, and 76.8% (PPI) according to GeniePath. The F1 difference for VRGCN is likely due to the experiment setting: for fair comparison, we used a fully-supervised setting for all SOTAs and LWGCN as FastGCN did, whereas the original VRGCN results were in a semi-supervised setting. **Q4: Applying Layer-Wise Training (LWT) on N-GCN (UAI'19).** *Reply:* This is a great suggestion. We will add the comparison and incorporate LWT into both components of DeepWalk and GCN of N-GCN.

▷ **Reviewer #4. Q: Math Confusions.** *Reply:* We appreciate and will strengthen the clarity of all mathematical notations. (1) $\mathcal{N}(i)$ is the set of node neighbors for the i^{th} node. (2) $x_{i,G_1}^{(l)}$ is the input of the i^{th} node in the graph G_1 at the l^{th} layer; and $x_{i,G_1,Con}^{(l)}$ and $x_{i,G_1,Lay}^{(l)}$ are the $x_{i,G_1}^{(l)}$ for conventional and layer-wise training, respectively. We will change "Con" and "Lay" to GCN and LWGCN, respectively; and we will add the page numbers in final version.

▷ **Reviewer #5. Q1: Search Time Exclusion.** *Reply:* Reported time in Table 4 was the training time and excluded the search time on hyper-parameters (will be reported in revision), as complexities in Table 1 were. Also, we did not include hyper-parameter (HP) tuning (e.g. grid search) time needed in other methods, as we compared to their trained models and could not access their search time. As correctly pointed out, a typical L2O in LWGCN can be more expensive than HP tuning in competitors. However, RNN controllers learned over (especially large) datasets are shown to be "transferrable" to other datasets; thus they do not have to be re-trained every time. We have experimentally verified the controller pre-trained on Amazon-3M can be plugged into other datasets, without further tuning, and achieve similar or close performance. Therefore, L2O can actually save time compared to dataset-specific manual tuning of HPs. Due to the rebuttal policy, we cannot include the additional results here but will definitely do so in revision. **Q2 & Q10: HPO Discussion and Comparison.** *Reply:* We thank the reviewer for the suggestion. We will include in revision. **Q3: Invalid Theoretical Analysis.** *Reply:* We agree that GCN layers are not necessarily injective (while readouts are) but respectfully disagree that the theory is invalid. In the theoretical setting laid out by GIN [4], parameterized MLP layers in GNN can approximate injective multiset functions thanks to the universal approximation theorem. Thms. 2 & 5 followed their setting. More importantly, Thm. 6 went beyond the setting to prove the power of LWGCN, even when layers are not injective, which is our major theoretical contribution. **Q4 & Q9: Effectiveness of Controller.** *Reply:* We will include the suggested curve. Meanwhile, we have to emphasize that the HP space in our L2O is of much lower dimension (stop or not) compared to other AutoML scenarios (e.g., NAS). Including the loss in the state is a widely adopted practice in L2O that we followed; e.g., see [5] for more rationales. **Q5, Q11, Q13: Figure/Table Confusions.** *Reply:* We agree on the suggestions on Fig. 1 & 4 and will revise accordingly. The stopping decision is made every 5 steps by the controller, which will be clarified for actions in Sec. 3.3. **Q6: Unpredictable Results.** *Reply:* We appreciate the point to additionally show the theoretical sanity of layer-wise training (LWT), besides what we showed through representation power (Thm. 6). Yet we have to disagree that LWT or intermediate loss "may bring unpredictable results". Empirically, it is commonly observed that layer-wise greedy pre-training improves the end task performance, even in latest models [1, 2]. The theoretical underpinnings remain challenging in CNNs and totally unexplored in GCNs. **Q7 & Q12: Insignificant Improvement.** *Reply:* **Time**: We did report up to a 4-layer GCN and will include much deeper ones during revision. Meanwhile, we emphasize the scalability to large, dense graphs (even with shallow GCNs) as a more pressing issue to resolve for GCN, which motivates the booming works for fast GCN training. Examples include recommendation systems [3] requiring nearly real-time updates of shallow GCNs. **Accuracy**: For the largest dataset Amazon-3M, LWGCN(-L2O) actually had comparable/better F1 to VRGCN while being faster (see also our response to Q1). Other competitors (GraphSAGE, FastGCN) cannot even been trained within a day, which will be made clear in the caption of Table 4. **Memory**: The complexities in Table 1 were for node propagation only and the same D for all methods. Reported in Table 4 was the GPU memory usage of the whole training. Such inconsistency will be corrected in revision. **Q8: Scalability.** *Reply:* In the conventional node classification setting, graph data are indeed loaded first. Compared to conventional training, LWT actually saves IO and storage since inputs to each layer are discarded after the layer is trained. Empirically we used Amazon-3M with millions of nodes, the largest public dataset (as R2 pointed out) in literature, yet did not observe a scalability issue. Image recognition is whole graph-level classification which is totally beyond the study's scope.

[1] E. Belilovsky et. al. Greedy layerwise learning can scale to imagenet. 2018. 1

[2] S. Löwe et al. Putting an end to end-to-end: Gradient-isolated learning of representations. In NeurIPS, 2019. 1

[3] B. Perozzi et al. Deepwalk: Online learning of social representations. 2014. 1

[4] K. Xu et al. How powerful are graph neural networks? ICLR, 2018. 1

[5] X. Zhen et al. Learning an adaptive learning rate schedule. NeurIPS, 2019. 1