We thank the reviewers for their feedback. We would like to note that due to space constraints, we relegated some theoretical and experimental details to the Appendix. We tried to ensure that the main paper, together with appendix, is self-contained. In Appendix B, we describe the greedy algorithms of Huang et al and Bengio et al. Appendix D contains a discussion of the theoretical results. Appendix J contains the experimental setup, hyper-parameter selection. In the final version of the paper, we will move additional clarifying details from the appendix to the main paper, as detailed in our responses below. Regarding XGBoost experiments, line 308 has a typo. We actually used decision trees (not decision stumps) in our experiments. The list of hyper-parameters tuned for XGBoost can be found in Table 7 in the appendix. We used trees with depth up to 20 in our experiments.

Reviewer 1. *Algorithms 2, 3.* We believe having both the algorithms can provide a unified view of various boosting algorithms: many classical boosting algorithms can be derived using Algorithm 3; whereas, greedy algorithms for DNN training can be derived using Algorithm 2. Due to space constraints, we didn't add experimental results comparing both the algorithms. In our experiments, we noticed Algorithm 3 has marginally worse performance than Algorithm 2.

Reviewer 2. *Novelty.* As pointed out in Section 3.1 and Appendix J.1, existing greedy techniques have certain drawbacks which cause them to perform poorly. One of the contributions of our work is to fix these drawbacks using better greedy techniques. While the resulting architecture might resemble ResNets, DenseNets, they have certain crucial differences which cause them to perform better.

Training NNs with boosting has already been proposed. Proposition 3.1 proves the non-obvious result that all these techniques are actually equivalent to greedy layer-by-layer technique of Bengio et al and hence face the drawbacks stated in Section 3.1. Moreover, our proposed techniques strictly improve upon the existing techniques.

Goal of our work. We would like to emphasize that the reason behind studying greedy algorithms is not to speed-up optimization of deep networks. Rather, the main reason is to come up with algorithms which have strong theoretical guarantees and at the same time have good empirical performance. Our generalized framework satisfies both the criteria (also see our response to Reviewer 3). Moreover, greedy algorithms come with certain additional advantages such as low memory requirements. For details on computation time, see our response to Reviewer 4.

Empirical Evaluation. Details about the hyper-parameters such as number of boosting iterations can be found in Appendix J.2. We used a hold-out set to choose the number of boosting iterations, with an upper limit of 15 iterations. For all the datasets, the optimal iteration picked using this process is less than 15. Lines 315-317 specify the types of weak feature transformers used. In Appendix J.1, we provide more experimental results showing why the proposed techniques work better than StdCompBoost. We believe our results clearly show that the proposed techniques achieve the above stated goals. On almost all the datasets, our techniques are better than StdCompBoost and other additive boosting techniques. On some datasets, we even match the performance of end-to-end training. Finally, thanks for references [1,2]; we will add more experimental results using datasets from these repositories.

Reviewer 3. *Related Work.* Thanks for pointing out [a,b], which we were unaware of. In [a], the authors propose a slight variant of AdaBoost in which the initial distribution is allowed to be any probability distribution, and the classification errors are measured using a general cost function. In [b], the authors consider an additive boosting procedure where different kinds of weak hypothesis classes are used in each iteration. Our work differs from [a,b] in a number of aspects, the key aspect being the way we aggregate weak classifiers. [a,b] consider additive combinations of weak classifiers, whereas we consider more general combinations. Moreover, unlike [a,b], the complexity of weak hypothesis classes in our algorithms grows with iteration.

Goal of our work. Our framework is a first step towards bridging the gap between classical methods such as boosting and modern deep learning methods. Classical boosting comes with strong theoretical guarantees, but doesn't match the performance of DNNs. Whereas, DNNs have good empirical performance, but don't come up with strong theoretical guarantees. Our proposed framework allows to derive learning algorithms that come with strong theoretical guarantees and improve the performance of existing greedy algorithms (see conclusion).

Comparison with end-to-end. In our experiments we did compare with ResNets: end-to-end training corresponds to ResNets trained using SGD. We would like to emphasize that our goal is not to beat end-to-end trained deep networks (see our claims in abstract lines 16-20 and introduction lines 65-70). While matching the performance of end-to-end training is the ideal goal, in this work we set out with a different goal: we aim to improve the performance of existing greedy techniques and take a step towards bridging the performance between greedy algorithms and end-to-end training.

Reviewer 4. *Theoretical guarantees.* For the theoretical guarantees to hold, we only need the feature transformers that are fit at each iteration to satisfy certain weak learning condition (Definition 4.1). This condition doesn't require the shallow networks to be trained optimally. This is similar to the weak learning condition in traditional boosting, which requires the weak learners added at each iteration to have better than random performance.

Computation. In terms of training time, we didn't notice any significant advantage for greedy techniques over end-to-end training. However, greedy techniques consume very little memory compared to end-to-end training. So greedy techniques can be very useful for training huge models on GPUs with very little memory. That being said, greedy techniques has several other advantages: they are easy to optimize and are theoretically much easier to analyze, as one only needs to understand optimization of shallow networks.