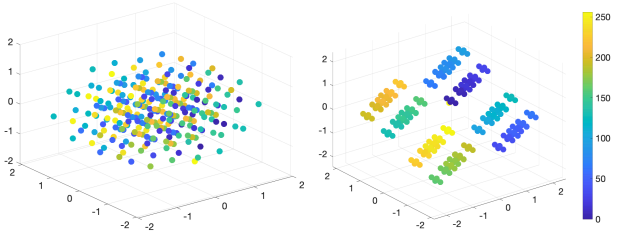
We thank the reviewers for detailed and constructive reviews and respond to each reviewer's questions:

R1: Utility of Strong Generalization: Generalization and strong generalization on out-of-distribution data (covariate shift) are fundamentally unsolved problems in both machine learning and specifically program synthesis/neural execution. Our insight that faithfully learned feedback masks helps strong generalization of transformers in a difficult domain is an important contribution to understanding why strong generalization occurs.

Importance of Strong Supervision: Our primary objective with this paper is to establish whether strong generalization can be achieved (to a reasonable degree) in a mechanistic sense. This is effectively a form of imitation learning. Followup work would focus on relaxing this constraint, but we believe that this is beyond the scope of a single paper. Other papers in the field have taken a similar path, for example [21] explores program synthesis via strong supervision and Pierrot et al., 2019 build on this with an RL solution called AlphaNPI that uses weak supervision.

[29] Comparison: Code for [29] is not available, making faithful reproduction difficult. NEE & [29] are based on different mechanisms. At each step, [29] requires supervision on every node and edge, while NEE instead supervises on the mask and nodes. We tried to match their setting (using the same graph types) and define our accuracy using a stricter metric (an exactly correct whole-graph result instead of next-node accuracy). For MST, [29]'s next-node accuracy at 100 node graphs is 41.37%, implying lower accuracy with the strict whole-graph correct metric vs close to 100% for NEE. There are evaluation differences ([29] trains using 20-node graphs, NEE trains on 8-node graphs), but we feel our evaluation vs. [29] is fair. [29] suffers from attention degradation which they try to counteract with max aggregation and entropy penalties.



(a) Before training (b) After training

Figure-R 1: PCA projections on number embeddings

Binary Ablation: We provide a binary ablation in the Appendix: (Fig.7, all_modifications variant V.S. all_modifications_one_hot_emb variant). **Figure-5 Insight:** We don't believe that it's expected for close numbers to necessarily show structure due to PCA. Figure-R 1 shows the embedding projections before and after training.

R2: Motivation: Briefly, we establish that, even with strong supervision, transformers can't generalize on simple algorithmic tasks. We propose a learned conditional masking mechanism as a fundamental technique to improve this. **Results:** We can't say that performance will be perfect in general, but we observed this behavior in our experiments. On the MST task, we do notice some degradation over large test graphs, but the degradation is graceful. **Figure-5:** We'll work on numerical analysis. Our goal was to establish that a) these embeddings have structure, b) these structures differ meaningfully across tasks, and c) non-observed numbers are embedded correctly into their appropriate spots.

R3: Baseline Transformer Masks: We apologize for not being clear here. Yes, both NEE and the baseline transformer have the same strong supervision granularity (output value & correct mask). During training, the baseline transformer is also given the ground truth masks (Figure 3). During testing, masks are not provided for both models and the baseline's attention mask consequently generalizes poorly (Figure 4). NEE's improvement is due to the learned mask and architectural design. A plain transformer without mask modifications at both training and testing times performs poorly (which is why we started working on NEE). We'll include this ablation in the next version of the paper.

Figure 4: The element in the (x, y) grid indicates the strength of the attention score between the xth number and the yth number. Figure-4 shows that the attention in the first 8 rows of the baseline transformer (below training data length, attention between the smallest 8 numbers) is clear, but out of distribution attention (beyond 8 numbers) grows fuzzy, resulting in incorrect predictions.

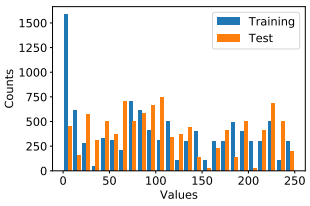


Figure-R 2: Histograms of data used in MST(Setting 2)

Performance: Yes, we find that using subroutines that strongly generalize, the resultant composition also strongly generalizes. In Table-1 we evaluate a challenging setting (training on 8-nodes, testing on 100). In two cases for shortest path, accuracy isn't exactly 100%, but 99.99% and 99.98%, which we don't consider significant deviations. In Table-5, using the learned subroutines we achieve 100% accuracy - scenario 1 (S1, first two table rows). You refer to the second two rows (S2), where we train using only Erdős-Rényi random graphs but test with a mix of 4 kinds of graphs (Section A.3, Appendix). In this case, the test distribution drifts from the training distribution (Figure-R. 2), and performance suffers, although the resultant accuracy is still much higher than [29].

Holding Out Numbers: Yes, the numbers are randomly selected. You are correct that if a bit is not toggled (i.e. bit 6 for 64 in your example), the model would not be able to generalize, but this motivates future work to create numerical basis that extrapolate across unseen bits, whereas our focus here is on interpolation.