Compositional Self-Improvement

Changho Shin [1] **Daiwei Chen** [2] **John Cooper** [3] **Brenden Lake** [1 4] **Frederic Sala** [3] **Ramya Korlakai Vinayak** [2]

Abstract

Self-improvement is increasingly important as many tasks become too complex for reliable human annotation. However, it typically fails when a model must solve harder out-of-domain instances in order to generate its own supervision. Many tasks, however, exhibit an underlying compositional structure. We show that when such a structure is present, this limitation can be overcome. Our key idea is to construct supervision for complex instances by composing predictions on smaller, in-distribution subproblems, a process we call *compositional self-improvement*. Iterating this process progressively expands the range of instances the model can solve reliably. We show, both empirically and theoretically, that this approach can bypass direct out-of-distribution generalization by reducing initially out-of-distribution instances to compositions of in-distribution subproblems. We further show that filtering out compositions prone to structured errors does not merely avoid hard cases: the resulting models can still generalize to the filtered-out slices. Compositional self-improvement provides a concrete path to scale beyond the original supervision regime.

1. Introduction

Self-improvement refers to learning from self-generated supervision, such as pseudo-labels or model-generated feedback (Lee, 2013; Huang et al., 2023; Lee et al., 2025). This is appealing because it can provide supervision even when humans cannot reliably annotate the task. However, extending such supervision beyond the original human-supervised regime requires the model to generate reliable pseudo-labels or feedback out of distribution. This is difficult because out-of-distribution generalization on more complex tasks remains challenging for modern models (Koh et al., 2021). How can we reliably extend self-improvement beyond the original supervision regime?

One way to address this challenge is to exploit structure in the task. Many tasks exhibit compositional structure: out-of-distribution instances can often be decomposed into compositions of in-distribution ones. This creates an opportunity to bypass unreliable self-generated supervision on out-of-distribution instances by constructing supervision from in-distribution subproblems.

We study this idea as *compositional self-improvement*. Rather than generating supervision directly on harder instances, the model constructs supervision indirectly by decomposing an instance into subproblems and combining their solutions through a task-specific composition rule. This reframes self-improvement as building supervision from reliable in-distribution components, instead of relying on direct out-of-distribution prediction.

Summary of Contributions.

1. We introduce *compositional self-improvement*, a self-training procedure that constructs pseudo-labels for harder instances by composing predictions on smaller, more reliable subproblems.
2. We instantiate this idea on tasks with explicit compositional structure, including run-length prediction, addition, and multiplication.
3. We show empirically that composition-based pseudo-labeling can expand the range of instances a model solves reliably, whereas direct pseudo-labeling fails outside the seed regime.
4. We identify structured composition errors as a key failure mode and show that filtering unsafe compositions can prevent systematic pseudo-label errors while preserving generalization to the filtered-out cases.

2. Setup and Method

We formalize compositional self-improvement in settings where larger instances can be constructed from smaller ones, and their outputs can be obtained by combining the outputs of those subproblems. Figure 1 illustrates the proce-

[1] Department of Computer Science, Princeton University, Princeton, NJ, USA [2] Department of Electrical and Computer Engineering, University of Wisconsin–Madison, Madison, WI, USA [3] Department of Computer Sciences, University of Wisconsin–Madison, Madison, WI, USA [4] Department of Psychology, Princeton University, Princeton, NJ, USA. Correspondence to: Changho Shin <cs1095@princeton.edu>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. *Copyright 2026 by the author(s).*

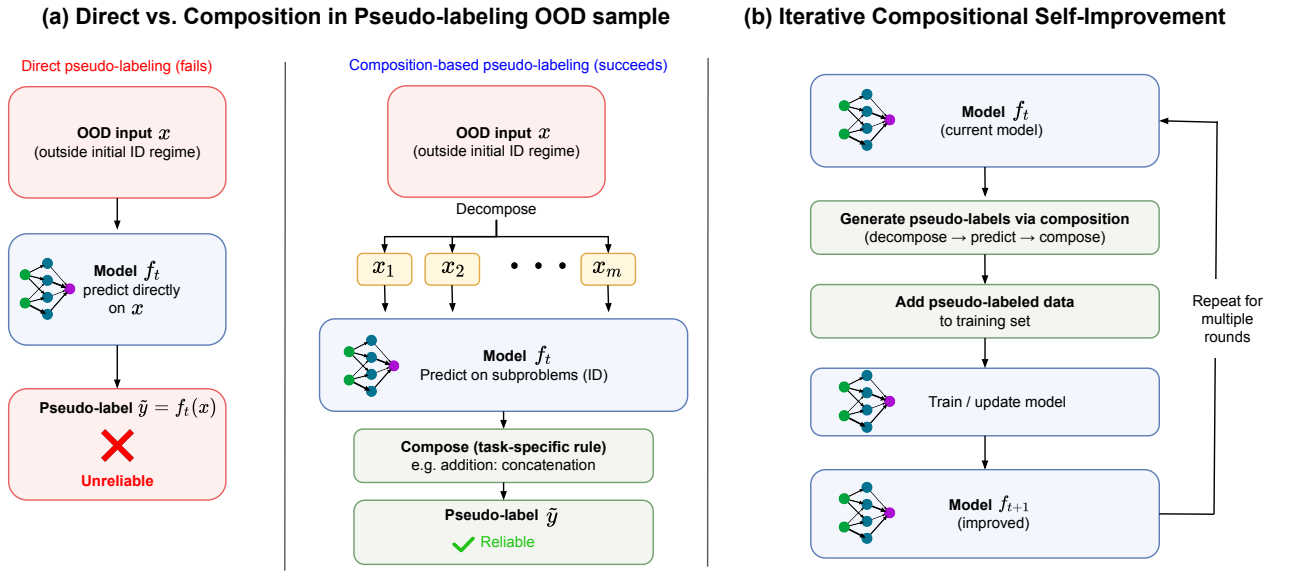


Figure 1. Overview of compositional self-improvement. Direct pseudo-labeling relies on the model's prediction on a harder out-of-distribution instance, where self-generated labels can be unreliable. Compositional self-improvement instead routes supervision through smaller in-distribution subproblems: the model predicts each subproblem, a task-specific rule composes the predictions into a pseudo-label, and the model is retrained on the resulting examples. Repeating this process progressively expands the model's reliable regime.

dure: rather than pseudo-labeling a harder instance directly, the model predicts smaller subproblems, composes their predicted outputs into a pseudo-label, and retrains on the resulting examples.

In the ideal case, for inputs x and x' in a composable regime,

$$f^\star(x \circ x') = f^\star(x) \diamond f^\star(x'),$$

where $\circ$ composes inputs and $\diamond$ combines outputs. For example, in addition, digit blocks can be concatenated together with their blockwise sums: from $11 + 22 = 33$ and $33 + 44 = 77$, we obtain $1133 + 2244 = 3377$, provided no carry crosses the block boundary.

We begin with an in-distribution seed set $S \subset X$ on which the model is reliable, and an initial model $\hat{f}_0$. Algorithm 1 summarizes the procedure. At each round, we sample $x_1, \ldots, x_m$ from the current pool, form $x = x_1 \circ \cdots \circ x_m$, and construct a pseudo-label by combining the model's predictions.

To handle imperfect composition, such as boundary carries in addition, we use a filtered aggregation rule:

$$\tilde{y} = \hat{\diamond}\big(\hat{f}_{t-1}(x_1), \ldots, \hat{f}_{t-1}(x_m)\big).$$

The rule either returns a composed pseudo-label or rejects the example, denoted $\tilde{y} = \perp$; rejected examples are discarded.

This procedure is useful when the composition rule produces sufficiently reliable pseudo-labels and enough composed examples are retained for retraining. Appendix B analyzes this tradeoff. We now evaluate the method empirically on tasks with explicit compositional structure.

Algorithm 1 Compositional self-improvement

Require: seed set S, initial model $\hat{f}_0$, aggregation rule $\hat{\diamond}$, rounds T

$\widetilde{D} \leftarrow \emptyset$, $D_0 \leftarrow S$
for $t = 1, \ldots, T$ **do**
  $\widetilde{D}_t \leftarrow \emptyset$
  for all $(x_1, \ldots, x_m)$ sampled from D_{t-1} **do**
    $x \leftarrow x_1 \circ \cdots \circ x_m$
    $\tilde{y} \leftarrow \hat{\diamond}(\hat{f}_{t-1}(x_1), \ldots, \hat{f}_{t-1}(x_m))$
    add $(x, \tilde{y})$ to $\widetilde{D}_t$ if $\tilde{y} \neq \perp$
  end for
  $\widetilde{D} \leftarrow \widetilde{D} \cup \widetilde{D}_t$
  $D_t \leftarrow D_{t-1} \cup \widetilde{D}_t$
  train $\hat{f}_t$ on $S \cup \widetilde{D}$
end for
return $\hat{f}_T$

3. Experiments

We evaluate compositional self-improvement on several classical algorithmic tasks with explicit logical structure, where larger, more complex instances can be constructed from smaller subproblems. Our experiments validate the following claims:

- **Expanding the solved regime:** The *Compositional Self-*

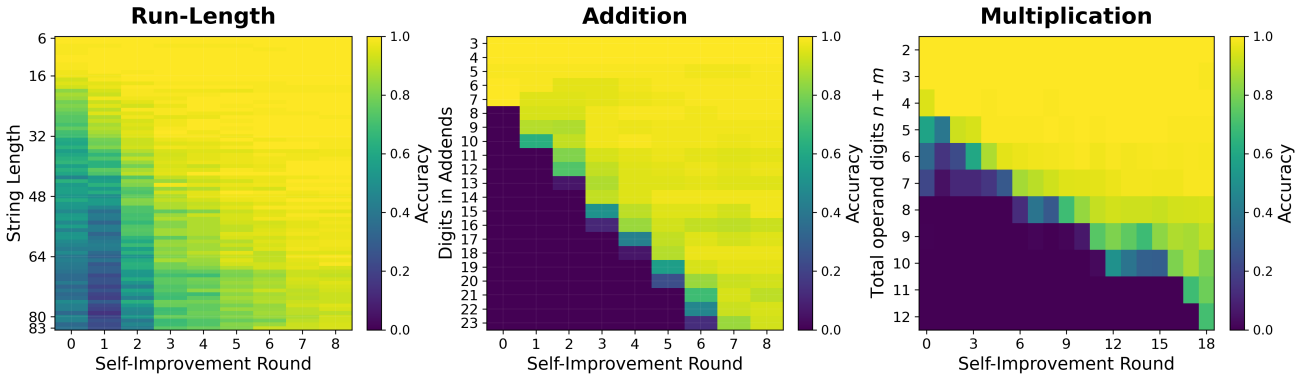


Figure 2. Compositional self-improvement expands the range of instances solved by the model. Each heatmap shows held-out accuracy across self-improvement rounds. The panels evaluate run-length prediction by string length, addition by addend digit length, and multiplication by total operand digits $n + m$. In all three tasks, reliable prediction extends beyond the seed model's reliable range.

Improvement method extends reliable prediction beyond the seed regime across all three tasks, with no observed plateau within the evaluated range (Section 3.1).

- **Controlling structured composition errors:** Composition provides more reliable pseudo-labels than direct prediction on harder instances, but unsafe compositions can introduce systematic errors (Section 3.2).
- **Generalizing beyond filtered examples:** Filtering unsafe compositions remove erroneous pseudo-labels from training, while the resulting model still surprisingly generalizes to held-out boundary-carry cases (Section 3.3).

Filtering applies only to composed training examples; evaluation still uses the full task distribution. Detailed setups, controlled synthetic experiments, and additional ablations are reported in Appendix C.

3.1. Compositional Self-Improvement Expands the Solved Regime

We first test whether compositional self-improvement can expand the range of instances a model solves reliably.

Setup. Across tasks, we start from a seed model reliable on a small in-distribution regime, then expand by composing pseudo-labels for larger instances and fine-tuning after each round. We evaluate three compositional tasks:

- *Run-length.* Predict the symbol and length of the leftmost longest run. For example, $00111 \mapsto 1|3$ and $2220 \mapsto 2|3$ compose to $001112220 \mapsto 1|3$ when the boundary symbols differ; compositions with possible boundary merges are filtered from training. The seed covers lengths 6–10, and eight rounds expand evaluation to length 83.
- *Addition.* Predict the sum of two decimal integers. For example, $11 + 22 = 33$ and $33 + 44 = 77$ compose to $1133 + 2244 = 3377$ when no carry crosses the block boundary; compositions with boundary carries are filtered from training. The seed covers 3–7 digit addends, and each round adds two digits up to 23.
- *Multiplication.* Predict products of decimal integers. We compose larger products using the standard digitwise multiplication algorithm: for example, 123×45 is built from $123 \times 5 = 615$ and $123 \times 4 = 492$, then shifted and added as $615 + 4920 = 5535$. The seed covers operands up to 2-by-2, and later rounds gradually introduce larger operand shapes up to 6-by-6; we report accuracy by the total operand digit count $n + m$.

Throughout the experiments, we treat a problem size as reliably solved when held-out accuracy exceeds 0.9.

Results. Figure 2 shows that compositional self-improvement expands the solved regime in all three tasks, reaching length 83 for run-length, 23-digit addends for addition, and strong multiplication performance through $n + m = 12$. Thus, composed predictions on smaller subproblems provide effective supervision for larger instances.

3.2. Composition Helps Only When Structured Errors Are Controlled

We next study composition under an imperfect rule. Addition provides a useful test case: its natural composition rule, blockwise concatenation, fails in the presence of boundary carries. This captures a broader challenge in compositional self-improvement, where simple composition rules may introduce structured errors.

Setup. Using the same seed regime and expansion schedule, we compare three pseudo-labeling strategies. *Direct* pseudo-labeling asks the current model to label the next addend-length range in one step. *Unfiltered* composition concatenates predicted blockwise sums without checking for boundary carries. *Filtered* composition discards boundary-carry cases before retraining. Additional task-level baselines are reported in Appendix C.

Results. Figure 3 shows that composition drives the

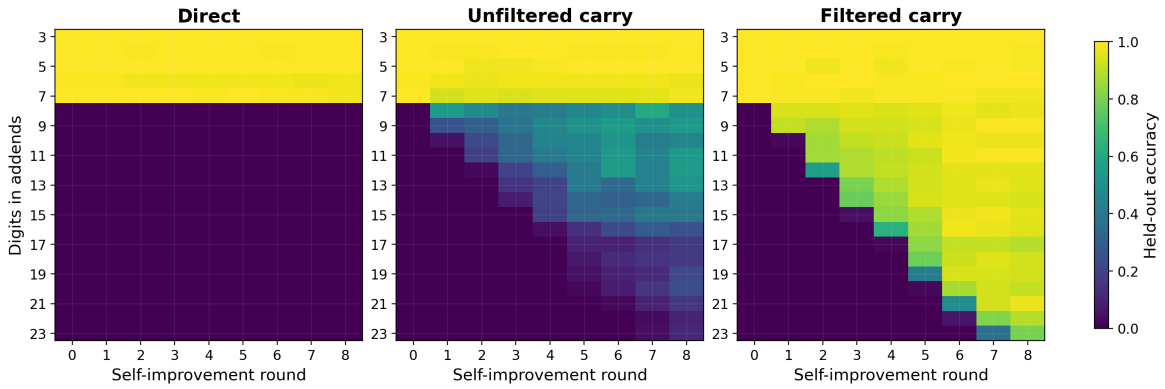


Figure 3. Addition baselines under the same expansion schedule. Direct pseudo-labeling fails outside the seed regime; unfiltered composition improves but injects boundary-carry errors; filtered composition yields stable expansion.

gain, but only when structured errors are controlled. Direct pseudo-labeling fails outside the seed regime, while unfiltered composition accumulates boundary-carry errors. Filtered composition removes this noise, reaching 0.965 final accuracy and 0.93 accuracy at 23 digits.

3.3. Filtered Composition Generalizes to Boundary-Carry Cases

Filtering removes cases where the imperfect rule is most likely to fail, thereby preventing structured pseudo-label errors. However, these cases remain part of the target task, and we would like the self-improved model to solve them rather than simply avoid them. We therefore ask whether the model can generalize to these filtered-out cases.

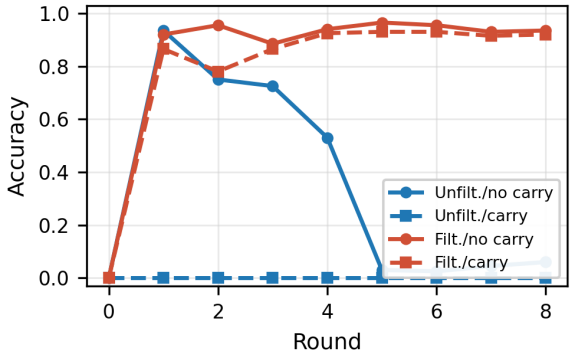


Figure 4. Addition accuracy on boundary slices. Color denotes the training procedure: unfiltered versus filtered composition. Line style denotes the test slice: with or without a middle-boundary carry. Filtered composition generalizes to both slices, while unfiltered composition degrades from boundary-carry errors.

Setup. We compose each example by splitting the addends into two equal-width digit blocks at the middle position. At each epoch, we evaluate on held-out examples with a fixed middle boundary, separating cases with and without a carry across that boundary. We compare the unfiltered and filtered composition on these slices.

Results. Figure 4 shows that filtering does not merely reduce pseudo-label error. Unfiltered composition fails on middle-boundary carry examples and later degrades even on no-carry cases. In contrast, filtered composition trains only on safe examples yet achieves high accuracy on both slices, showing generalization beyond the filtered subset.

4. Related Work

We discuss the most relevant related work here; extended discussion appears in Appendix A.

Compositional generalization. Compositional generalization studies whether models can extrapolate from seen combinations to unseen ones (Lake & Baroni, 2018; Keysers et al., 2020; Kim & Linzen, 2020). Prior work analyzes distinct failure modes (Hupkes et al., 2020) and improves systematic generalization through architectural bias, neuro-symbolic structure, or scale and coverage (Bahdanau et al., 2019; Liu et al., 2020; Chen et al., 2020; Redhardt et al., 2025). We shift the burden from generalization to construction by using composition to systematically reduce out-of-domain instances to in-domain subproblems, rather than relying on a model's ability to extrapolate compositionally.

Self-improvement. Self-improvement methods train models from their own outputs, rationales, critiques, or preferences (Zelikman et al., 2022; Madaan et al., 2023; Zelikman et al., 2024; Scheurer et al., 2022; Yuan et al., 2024). Recent iterative self-training methods improve easy-to-hard or length generalization in algorithmic domains (Huang et al., 2023; Lee et al., 2025). In contrast, we reduce reliance on direct predictions at the edge of the model's reliable regime: composition turns predictions on smaller subproblems into supervision for larger instances.

5. Conclusion

We introduced compositional self-improvement, which leverages task structure to construct pseudo-labels for larger instances from smaller, reliable subproblems. Across three classical logically compositional tasks: run-length prediction, addition, and multiplication, the method expands reliable prediction beyond the seed regime and outperforms direct pseudo-labeling. These results show that composition can turn self-improvement from direct, unreliable extrapolation to controlled data-construction process, allowing models to move beyond their original supervision range when reliable composition rules are available.

Limitations and Future Directions. Our experiments focus on algorithmic tasks with explicit, hand-designed composition rules, where decomposition and aggregation can be specified cleanly. Extending compositional self-improvement to more realistic domains remains an important direction. In many settings, composition may be approximate, latent, or learned from data rather than provided a priori. Future work could study whether models can discover useful decomposition and composition strategies automatically, for example through program synthesis, agentic search, or learned verifiers.

References

Angluin, D. and Laird, P. D. Learning from noisy examples. *Machine Learning*, 2(4):343–370, 1988. doi: 10.1023/A:1022873112823.

Bahdanau, D., Murty, S., Noukhovitch, M., Nguyen, T. H., de Vries, H., and Courville, A. Systematic generalization: What is required and can it be learned? In *International Conference on Learning Representations*, 2019.

Bai, Y. et al. Constitutional AI: Harmlessness from AI feedback, 2022. arXiv preprint arXiv:2212.08073.

Chen, X., Liang, C., Yu, A. W., Song, D., and Zhou, D. Compositional generalization via neural-symbolic stack machines. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1690–1701, 2020.

Chen, Z., Deng, Y., Yuan, H., Ji, K., and Gu, Q. Self-play fine-tuning converts weak language models to strong language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6621–6642. PMLR, 2024.

Dou, Z.-Y., Yang, C.-F., Wu, X., Chang, K.-W., and Peng, N. Re-ReST: Reflection-reinforced self-training for language agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 15394–15411, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.861.

Hu, C., Hu, Y., Cao, H., Xiao, T., and Zhu, J. Teaching language models to self-improve by learning from language feedback. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6090–6101, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.364.

Huang, J., Gu, S., Hou, L., Wu, Y., Wang, X., Yu, H., and Han, J. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1051–1068, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.67.

Hupkes, D., Dankers, V., Mul, M., and Bruni, E. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67: 757–795, 2020. doi: 10.1613/jair.1.11674.

Ito, T., Klinger, T., Schultz, D., Murray, J., Cole, M., and Rigotti, M. Compositional generalization through abstract representations in human and artificial neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 32225–32239, 2022.

Keysers, D., Schärli, N., Scales, N., Buisman, H., Furrer, D., Kashubin, S., Momchev, N., Sinopalnikov, D., Stafiniak, L., Tihon, T., Tsarkov, D., Wang, X., van Zee, M., and Bousquet, O. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2020.

Kim, N. and Linzen, T. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 9087–9105, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.731.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B., Haque, I., Beery, S. M., Leskovec, J., Kundaje, A., Pierson, E., Levine, S., Finn, C., and Liang, P. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5637–5664. PMLR, 2021.

Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2873–2882. PMLR, 2018.

Lake, B. M. and Baroni, M. Human-like systematic generalization through a meta-learning neural network. *Nature*, 623(7985):115–121, 2023. doi: 10.1038/s41586-023-06668-3.

Lee, D.-H. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013.

Lee, N., Cai, Z., Schwarzschild, A., Lee, K., and Papailiopoulos, D. Self-improving transformers overcome easy-to-hard and length generalization challenges. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 32930–32964. PMLR, 2025.

Liu, Q., An, S., Lou, J.-G., Chen, B., Lin, Z., Gao, Y., Zhou, B., Zheng, N., and Zhang, D. Compositional generalization by learning analytical expressions. In *Advances in Neural Information Processing Systems*, volume 33, pp. 11416–11427, 2020.

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B. P., Hermann, K., Welleck, S., Yazdanbakhsh, A., and Clark, P. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46534–46594, 2023.

Massart, P. and Nédélec, É. Risk bounds for statistical learning. *The Annals of Statistics*, 34(5):2326–2366, 2006. doi: 10.1214/009053606000000786.

Redhardt, F., Akram, Y., and Schug, S. Scaling can lead to compositional generalization. In *Advances in Neural Information Processing Systems*, 2025. Spotlight.

Scheurer, J., Campos, J. A., Chan, J. S., Chen, A., Cho, K., and Perez, E. Training language models with language feedback, 2022.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. ISBN 9781107057135.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023.

Yuan, W., Pang, R. Y., Cho, K., Li, X., Sukhbaatar, S., Xu, J., and Weston, J. E. Self-rewarding language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57905–57923. PMLR, 2024.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. D. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 15476–15488, 2022.

Zelikman, E., Harik, G., Shao, Y., Jayasiri, V., Haber, N., and Goodman, N. D. Quiet-STaR: Language models can teach themselves to think before speaking. In *Proceedings of the First Conference on Language Modeling*, 2024.

Appendix

Appendix A expands the discussion of related work, situating our approach within compositional generalization and self-improvement. Appendix B presents the theoretical analysis, including bounds on pseudo-label noise, conditions for learning under noisy composed supervision, and a comparison to direct composition. Appendix C contains experimental details, including task-specific composition rules, synthetic experiments varying noise and sample size, additional baselines, and ablations on seed strength and composed data size.

A. Extended Related Work

A.1. Compositional Generalization

Compositional generalization asks whether a model can reuse learned primitives and rules in combinations not observed during training. Lake & Baroni (2018) study this question with SCAN, a controlled sequence transduction benchmark, while Keysers et al. (2020) and Kim & Linzen (2020) extend it to more realistic semantic parsing benchmarks, CFQ and COGS. Hupkes et al. (2020) further argue that compositionality should not be treated as a single property, and propose separate tests for systematic recombination, productivity, locality, substitutivity, and overgeneralization.

A large body of work tries to improve compositional generalization by changing the model or training setup. Modular and neuro-symbolic approaches bias the learner toward operations that can be reused compositionally (Bahdanau et al., 2019; Liu et al., 2020; Chen et al., 2020). Other work studies the role of abstract representations, meta-learning, and scale or data coverage in producing systematic behavior (Ito et al., 2022; Lake & Baroni, 2023; Redhardt et al., 2025). These approaches primarily ask whether the model itself can extrapolate to unseen compositions.

Our focus is different. Rather than treating compositional structure as a target for evaluating structured out-of-distribution generalization, we use it as a mechanism for constructing supervision. This shifts the burden from generalization to construction: larger out-of-distribution in-

stances are reduced to smaller subproblems closer to the model's reliable regime, and their predictions are composed into pseudo-labels for the larger examples.

A.2. Self-Improvement and Self-Generated Supervision

Self-improvement methods train or guide models using signals produced by the model itself or by another AI system. In reasoning, STaR bootstraps chain-of-thought rationales by retaining generations that lead to correct answers (Zelikman et al., 2022). Quiet-STaR extends this idea toward token-level latent reasoning during continued pretraining (Zelikman et al., 2024). Other methods use feedback at inference time rather than additional training: Self-Refine iteratively generates, critiques, and revises an output using the same model (Madaan et al., 2023). In alignment and preference learning, language or AI feedback can also serve as supervision, as in language-feedback training, Constitutional AI, self-rewarding language models, and Self-Refinement Tuning (Scheurer et al., 2022; Bai et al., 2022; Yuan et al., 2024; Hu et al., 2024).

A related line studies iterative training and agent settings. Huang et al. (2023) fine-tune on high-confidence self-generated rationales. SPIN uses self-play fine-tuning to improve a language model without additional human annotations (Chen et al., 2024). Reflexion stores verbal reflections as memory for future decisions (Shinn et al., 2023), and Re-ReST improves low-quality agent trajectories with a reflector before using them for self-training (Dou et al., 2024). Across these methods, progress depends on whether the self-generated signal is reliable enough to train on.

The closest work to ours is Lee et al. (2025), who study iterative self-improvement for easy-to-hard and length generalization on algorithmic tasks. Their method improves the reliability of frontier pseudo-labels by sampling multiple solutions and applying majority voting or filtering. However, they also note that selecting reliable out-of-distribution examples remains challenging. Our method addresses this issue by using task structure to construct pseudo-labels: larger instances are labeled by composing predictions on smaller, more reliable subproblems. The model therefore need not solve the full larger instance before that instance enters training. The remaining challenge is that some compositions are invalid and can introduce structured pseudo-label errors, which motivates the filtered and unfiltered composition variants studied in our experiments.

B. Theoretical Analysis

This appendix gives a simple sufficient condition under which composed pseudo-labels support self-improvement. The analysis tracks three quantities: the component error of the current model, the error of the aggregation rule when supplied with correct component labels, and the number of accepted pseudo-labeled examples. The main conclusion is that composed labels need not be perfect. If their label noise is bounded away from $1/2$ on the accepted distribution, standard learning guarantees imply that retraining can learn the target classifier on that distribution.

Setup. Let $\mathcal{X}$ be the input space, $\mathcal{Y}$ the label space, and

$$f^\star : \mathcal{X} \to \mathcal{Y}$$

the target function. At round t, the current model is $\hat{f}_{t-1}$. A composition step samples component inputs

$$X_1, \dots, X_m$$

from the current pool and forms

$$X := X_1 \circ \cdots \circ X_m.$$

An aggregation rule combines the predicted component labels into a pseudo-label for the composed input:

$$\widehat{\diamond} : \mathcal{Y}^m \to \mathcal{Y} \cup \{\perp\},$$

where $\perp$ denotes rejection. In practice, $\widehat{\diamond}$ may also depend on filtering rules applied to the inputs (e.g., boundary conditions). We omit this dependence, as the analysis only uses the accepted pseudo-labels and their error rate. The tentative pseudo-label is

$$\widehat{\diamond}\big(\hat{f}_{t-1}(X_1), \dots, \hat{f}_{t-1}(X_m)\big).$$

If this value is $\perp$, the example is discarded. Otherwise, the example is accepted and we write

$$\widetilde{Y}_t := \widehat{\diamond}\big(\hat{f}_{t-1}(X_1), \dots, \hat{f}_{t-1}(X_m)\big) \in \mathcal{Y}.$$

Let $\mathcal{D}_t^{\mathrm{acc}}$ denote the distribution of accepted composed inputs X at round t. In the pseudo-label noise analysis below, all probabilities are taken over accepted composed examples. Equivalently, the distribution has already been restricted to the examples that pass the filter.

B.1. Noise of constructed pseudo-labels

We first bound the error rate of the pseudo-labels produced by the composition step. There are two sources of error. First, the model may predict one of the component labels incorrectly. Second, even if the component labels are correct, the aggregation rule may fail to produce the correct label for the composed input. We refer to the latter as the oracle aggregation error, since it is the error of the aggregation rule when evaluated on the true component labels.

Assume that the current model has component error at most ϵ_t on accepted compositions:

$$\Pr[\hat{f}_{t-1}(X_i) \neq f^\star(X_i)] \le \epsilon_t \qquad \text{for each } i.$$

Assume also that the oracle aggregation error is at most $\xi_{t,m}$:

$$\Pr[\widehat{\diamond}(f^\star(X_1), \dots, f^\star(X_m)) \neq f^\star(X)] \leq \xi_{t,m}.$$

Thus $\xi_{t,m}$ captures the error due to the aggregation rule itself, after removing component prediction errors from consideration.

Proposition B.1 (Noise of a composed pseudo-label)**.** *The accepted pseudo-label satisfies*

$$\rho_{t,m} := \Pr[\widetilde{Y}_t \neq f^\star(X)] \leq m\epsilon_t + \xi_{t,m}.$$

Proof. Let

$$E_i := \{\hat{f}_{t-1}(X_i) \neq f^\star(X_i)\}$$

be the event that component i is predicted incorrectly. Let

$$R := \{\widehat{\diamond}(f^\star(X_1), \dots, f^\star(X_m)) \neq f^\star(X_1 \circ \cdots \circ X_m)\}$$

be the event that the aggregation rule fails on the true component labels. If no event E_i occurs, then

$$\widetilde{Y}_t = \widehat{\diamond}(f^\star(X_1), \dots, f^\star(X_m)).$$

If R also does not occur, this is equal to $f^\star(X)$. Hence a pseudo-label error can occur only if some component prediction is wrong or the aggregation rule fails:

$$\{\widetilde{Y}_t \neq f^\star(X)\} \subseteq \bigcup_{i=1}^{m} E_i \cup R.$$

Taking probabilities and applying the union bound gives

$$\Pr[\widetilde{Y}_t \neq f^\star(X)] \leq \sum_{i=1}^{m} \Pr[E_i] + \Pr[R] \leq m\epsilon_t + \xi_{t,m}.$$

□

Proposition B.1 gives the pseudo-label noise bound used in the rest of the analysis. We now combine this bound with a standard finite-class guarantee for learning from noisy labels.

B.2. Finite-class noisy-label learning

We next use a standard finite-class argument for ERM with noisy labels under zero-one loss (Angluin & Laird, 1988; Massart & Nédélec, 2006; Shalev-Shwartz & Ben-David, 2014). We assume a uniform bound on the pseudo-label error:

$$\Pr[\widetilde{Y} \neq f^\star(X) \mid X = x] \leq \bar{\rho} < \tfrac{1}{2}.$$

Equivalently, the accepted pseudo-label is correct with probability at least $1 - \bar{\rho} > 1/2$ at every input. Under this condition, $f^\star$ has smaller pseudo-label risk than any classifier that disagrees with it; Lemma B.3 quantifies this gap.

Lemma B.2 (Uniform convergence for finite classes)**.** *Let $\mathcal{H}$ be finite, and let $\ell(h, z) \in [0, 1]$ for all $h \in \mathcal{H}$ and examples z. Let Z be a random example and let $z_1, \dots, z_n$ be i.i.d. draws from the distribution of Z. Then, with probability at least $1 - \delta$,*

$$\sup_{h \in \mathcal{H}} \left| L(h) - \widehat{L}_n(h) \right| \leq \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2n}},$$

where

$$L(h) := \mathbb{E}[\ell(h, Z)], \qquad \widehat{L}_n(h) := \frac{1}{n} \sum_{i=1}^{n} \ell(h, z_i).$$

Proof. For each fixed $h \in \mathcal{H}$, Hoeffding's inequality gives

$$\Pr\left[\left| L(h) - \widehat{L}_n(h) \right| > u\right] \leq 2e^{-2nu^2}.$$

Therefore, by the union bound,

$$\Pr\left[\sup_{h \in \mathcal{H}} \left| L(h) - \widehat{L}_n(h) \right| > u\right] \leq 2|\mathcal{H}|e^{-2nu^2}.$$

Taking

$$u = \sqrt{\frac{\log|\mathcal{H}| + \log(2/\delta)}{2n}}$$

makes the right-hand side equal to δ, which proves the claim. □

Lemma B.3 (Pseudo-risk comparison)**.** *Let $\mathcal{Y}$ be an arbitrary output space and write*

$$\rho(x) := \Pr[\widetilde{Y} \neq f^\star(X) \mid X = x].$$

For any classifier h, define the target risk

$$R(h) := \Pr[h(X) \neq f^\star(X)]$$

and the pseudo-label risk

$$\widetilde{R}(h) := \Pr[h(X) \neq \widetilde{Y}].$$

Then

$$\widetilde{R}(h) - \widetilde{R}(f^\star) \geq \mathbb{E}\left[(1 - 2\rho(X))\mathbf{1}\{h(X) \neq f^\star(X)\}\right].$$

In particular, if $\rho(x) \leq \bar{\rho} < 1/2$ for all x, then

$$\widetilde{R}(h) - \widetilde{R}(f^\star) \geq (1 - 2\bar{\rho})R(h).$$

Proof. Fix x. If $h(x) = f^\star(x)$, then h and $f^\star$ incur the same pseudo-label loss at x. Now suppose $h(x) \neq f^\star(x)$. Then

$$\begin{aligned}\Pr[h(x) \neq \widetilde{Y} \mid X = x] &- \Pr[f^\star(x) \neq \widetilde{Y} \mid X = x] \\ &= \Pr[\widetilde{Y} = f^\star(x) \mid X = x] - \Pr[\widetilde{Y} = h(x) \mid X = x].\end{aligned}$$

The first term is $1 - \rho(x)$. Since $h(x) \neq f^\star(x)$, the second term is at most $\rho(x)$. Therefore the conditional difference is at least

$$1 - 2\rho(x).$$

Averaging over X gives the first inequality, and the second follows from $\rho(x) \leq \bar{\rho}$. □

Theorem B.4 (Finite-class ERM under bounded pseudo-label noise)**.** *Fix $\delta \in (0,1)$. Let $\mathcal{D}_t^{\mathrm{acc}}$ be the accepted composed-input distribution at round t, and let $\mathcal{H}_t$ be a finite hypothesis class containing $f^\star$ on this distribution. Let $\hat{h}_t$ be an empirical risk minimizer on n_t i.i.d. accepted pseudo-labeled examples. Assume*

$$\Pr[\widetilde{Y}_t \neq f^\star(X) \mid X = x] \leq \bar{\rho}_t < \tfrac{1}{2} \qquad \textit{for all } x.$$

Then, writing

$$R_t(h) := \Pr_{X \sim \mathcal{D}_t^{\mathrm{acc}}}[h(X) \neq f^\star(X)],$$

with probability at least $1 - \delta$,

$$R_t(\hat{h}_t) \leq \frac{2}{1 - 2\bar{\rho}_t} \sqrt{\frac{\log|\mathcal{H}_t| + \log(2/\delta)}{2n_t}}.$$

Proof. Apply Lemma B.2 to the pseudo-label loss

$$\ell_h(X, \widetilde{Y}) = \mathbf{1}\{h(X) \neq \widetilde{Y}\}.$$

With probability at least $1 - \delta$,

$$\sup_{h \in \mathcal{H}_t} |\widetilde{R}(h) - \widehat{\widetilde{R}}(h)| \leq \nu_t, \qquad \nu_t := \sqrt{\frac{\log|\mathcal{H}_t| + \log(2/\delta)}{2n_t}}.$$

Since $\hat{h}_t$ minimizes empirical pseudo-risk and $f^\star \in \mathcal{H}_t$,

$$\widetilde{R}(\hat{h}_t) \leq \widetilde{R}(f^\star) + 2\nu_t.$$

By Lemma B.3,

$$(1 - 2\bar{\rho}_t) R_t(\hat{h}_t) \leq \widetilde{R}(\hat{h}_t) - \widetilde{R}(f^\star) \leq 2\nu_t.$$

Rearranging proves the theorem. □

To combine this result with the pseudo-label noise bound, we use the conditional form of Proposition B.1. If, for every accepted composed input $X = x$, the component error is at most ϵ_t and the aggregation error on true component labels is at most $\xi_{t,m}$, then the same union-bound argument gives

$$\Pr[\widetilde{Y}_t \neq f^\star(X) \mid X = x] \leq m\epsilon_t + \xi_{t,m}.$$

Thus an m-component construction satisfies the theorem whenever

$$m\epsilon_t + \xi_{t,m} < \tfrac{1}{2}.$$

If

$$m\epsilon_t + \xi_{t,m} \leq \tfrac{1}{2} - \gamma$$

for some $\gamma > 0$, then error at most α at round t is guaranteed whenever

$$n_t \geq \frac{\log|\mathcal{H}_t| + \log(2/\delta)}{2\gamma^2\alpha^2}. \tag{1}$$

Thus composed labels need not be noiseless. It is sufficient that accepted pseudo-labels are conditionally correct with probability at least $1/2 + \gamma$ at each input, and that enough accepted examples are retained.

Remark on the noise assumption. The bounded-noise condition is a sufficient condition on the accepted pseudo-labeled distribution, not on the raw model errors. It does not require errors to be independent, identically distributed, or symmetric. In practice, errors may be highly structured, depending on the task and aggregation rule; filtering removes compositions with known systematic errors before retraining. The theory abstracts these effects through the resulting bound on $\bar{\rho}_t$. This assumption need not hold uniformly before filtering, but is required only for the retained examples.

B.3. Iterating the one-step guarantee

We now iterate the one-step guarantee across rounds. The induction requires a compatibility condition between consecutive rounds: the components used to construct round-t examples are drawn from a distribution on which the round-$(t-1)$ error guarantee applies. This holds, for example, when the pool for round t is sampled from accepted examples produced at round $t - 1$.

Let m_t be the number of components used at round t. Define

$$r_t(e) := m_t e + \xi_{t,m_t}.$$

By Proposition B.1, $r_t(e)$ upper-bounds the accepted pseudo-label noise at round t whenever the model used to label each component has error at most e on the corresponding component distribution.

Lemma B.5 (One-step guarantee)**.** *Suppose that at round t the accepted pseudo-label noise satisfies*

$$\bar{\rho}_t \leq \tfrac{1}{2} - \gamma_t$$

for some $\gamma_t > 0$. If

$$n_t \geq \frac{\log|\mathcal{H}_t| + \log(2/\delta_t)}{2\gamma_t^2\alpha^2},$$

then the empirical risk minimizer $\hat{h}_t$ trained on the accepted round-t pseudo-labels satisfies

$$\Pr_{X \sim \mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X) \neq f^\star(X)] \leq \alpha$$

with probability at least $1 - \delta_t$.

Proof. Apply Theorem B.4 with failure probability δ_t. With probability at least $1-\delta_t$,

$$\Pr_{X\sim\mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X)\neq f^\star(X)] \le \frac{2}{1-2\bar{\rho}_t}\sqrt{\frac{\log|\mathcal{H}_t|+\log(2/\delta_t)}{2n_t}}.$$

Since $\bar{\rho}_t \le 1/2-\gamma_t$, we have

$$1-2\bar{\rho}_t \ge 1-2\left(\tfrac{1}{2}-\gamma_t\right)=2\gamma_t.$$

Therefore

$$\frac{2}{1-2\bar{\rho}_t}\le\frac{1}{\gamma_t},$$

and hence

$$\Pr_{X\sim\mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X)\neq f^\star(X)] \le \frac{1}{\gamma_t}\sqrt{\frac{\log|\mathcal{H}_t|+\log(2/\delta_t)}{2n_t}}.$$

The assumed lower bound on n_t implies

$$\frac{\log|\mathcal{H}_t|+\log(2/\delta_t)}{2n_t}\le\gamma_t^2\alpha^2.$$

Substituting this into the previous inequality gives

$$\Pr_{X\sim\mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X)\neq f^\star(X)] \le \frac{1}{\gamma_t}\cdot\gamma_t\alpha=\alpha.$$

This proves the claim. □

Proposition B.6 (Roundwise self-improvement)**.** *Fix a target error $\alpha>0$, failure probability $\delta\in(0,1)$, and number of rounds $T\ge 1$. Suppose the seed model used at round 1 has component error at most ϵ_1. Assume there exist margins $\gamma_1,\dots,\gamma_T>0$ such that*

$$r_1(\epsilon_1)\le\tfrac{1}{2}-\gamma_1$$

and, for every $t=2,\dots,T$,

$$r_t(\alpha)\le\tfrac{1}{2}-\gamma_t.$$

Assume also that, for each $t\ge 2$, conditional on the success of round $t-1$, the component inputs used to construct accepted round-t examples are drawn from distributions on which $\hat{h}_{t-1}$ has target error at most α. If, for every $t=1,\dots,T$,

$$n_t\ge\frac{\log|\mathcal{H}_t|+\log(2T/\delta)}{2\gamma_t^2\alpha^2},$$

then with probability at least $1-\delta$,

$$\Pr_{X\sim\mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X)\neq f^\star(X)]\le\alpha \qquad \textit{for all } t=1,\dots,T.$$

Proof. Set $\delta_t=\delta/T$. Let

$$G_t:=\left\{\Pr_{X\sim\mathcal{D}_t^{\mathrm{acc}}}[\hat{h}_t(X)\neq f^\star(X)]\le\alpha\right\}$$

be the event that round t succeeds.

For the first round, Proposition B.1 and the assumption $r_1(\epsilon_1)\le 1/2-\gamma_1$ imply that the accepted pseudo-label noise is at most $1/2-\gamma_1$. The sample-size condition therefore allows us to apply Lemma B.5 with failure probability δ/T, giving

$$\Pr(G_1^c)\le\delta/T.$$

Now fix $t\ge 2$ and condition on the success of the previous rounds. By the compatibility assumption, the components used at round t are drawn from a distribution on which the previous model has error at most α. Hence Proposition B.1 gives accepted pseudo-label noise at most

$$r_t(\alpha)\le\tfrac{1}{2}-\gamma_t.$$

Applying Lemma B.5 again gives

$$\Pr(G_t^c\mid G_1,\dots,G_{t-1})\le\delta/T.$$

Thus each round fails with conditional probability at most δ/T given that the earlier rounds succeeded. A union bound over the first failed round then gives

$$\Pr\left[\bigcap_{t=1}^{T}G_t\right]\ge 1-\sum_{t=1}^{T}\frac{\delta}{T}=1-\delta.$$

On this event, the desired error bound holds for all rounds $t=1,\dots,T$. □

The proposition shows how the one-step guarantee can be reused across an expanding sequence of accepted distributions. The first round requires the seed model to make sufficiently accurate component predictions so that $r_1(\epsilon_1)<1/2$. Later rounds require the retrained model's error level α to keep the next round's accepted pseudo-label noise below $1/2$. Thus retraining controls the component error before the next expansion step.

Comparison to direct composition. The guarantee above implies that, under the stated margin and sample-size conditions,

$$\Pr_{X\sim\mathcal{D}_T^{\mathrm{acc}}}[\hat{h}_T(X)=f^\star(X)]\ge 1-\alpha$$

with probability at least $1-\delta$. More generally, the same bound holds simultaneously for every round $t=1,\dots,T$.

This guarantee is obtained by controlling the pseudo-label noise locally at each expansion step. A direct-composition

baseline instead applies the seed model through the entire composition without intermediate retraining. After T rounds, a composed example depends on

$$K_T := \prod_{t=1}^{T} m_t$$

primitive components, so the corresponding union-bound error term scales as $K_T \epsilon_1$, before accounting for any aggregation error. The roundwise procedure replaces this global requirement with the local requirements

$$r_1(\epsilon_1) < \tfrac{1}{2}, \qquad r_t(\alpha) = m_t \alpha + \xi_{t,m_t} < \tfrac{1}{2} \quad (t \geq 2).$$

This gives a final accepted-distribution accuracy guarantee of at least $1 - \alpha$. Intermediate retraining passes a controlled error level α to the next round, rather than carrying the seed error through one long composition.

C. Experimental Details and Additional Results

C.1. Task-specific composition rules

We use three algorithmic tasks whose labels admit exact or filtered composition.

Run-length. Let $\Sigma = \{0, \ldots, 9\}$ be the symbol alphabet. For a string $x \in \Sigma^n$, a run is a maximal interval of equal symbols. Let

$$L(x) := \text{maximum run length in } x.$$

We use leftmost tie-breaking: if multiple runs attain length $L(x)$, let $i^\star(x)$ be the starting index of the earliest such run and let

$$a(x) := x_{i^\star(x)}.$$

The model target is the plain output pair

$$f^\star_{\text{run}}(x) := \big(a(x), L(x)\big),$$

serialized as `symbol|length`. For pairwise concatenation $x \circ y := xy$, the filtered composition rule accepts only when the boundary cannot merge two runs:

$$G(x, y) = \mathbf{1}\{x_{|x|} \neq y_1\}.$$

When $G(x, y) = 1$, no run crosses the boundary, so the longest run in xy is the longer of the two component longest runs:

$$f^\star_{\text{run}}(xy) = \begin{cases} f^\star_{\text{run}}(x), & L(x) \geq L(y), \\ f^\star_{\text{run}}(y), & L(y) > L(x). \end{cases}$$

The first case includes ties because every run in x appears before every run in y. When $G(x, y) = 0$, a cross-boundary run may be longer than either component prediction reveals, so the filtered rule rejects the composed example.

Addition. An addition example is a pair of decimal integers (A, B), and the target is the exact sum

$$f^\star_{\text{add}}(A, B) := A + B.$$

Suppose (A, B) is decomposed into decimal blocks

$$A = a_1 \| a_2 \| \cdots \| a_k, \qquad B = b_1 \| b_2 \| \cdots \| b_k,$$

where block i has width d_i digits and $\|$ denotes decimal concatenation. Let

$$s_i := a_i + b_i.$$

The composed input is obtained by concatenating the blocks of the two addends. The filtered composition rule accepts only when no carry propagates across any block boundary, i.e.,

$$s_i < 10^{d_i} \qquad \text{for every non-leading block } i = 2, \ldots, k,$$

equivalently, no less-significant block sends a carry into the next more-significant block. On accepted examples, the full sum is exactly the concatenation of the blockwise sums $s_1, \ldots, s_k$, written in their original block widths. The unfiltered addition baseline applies the same blockwise aggregation without rejection, which becomes noisy precisely when a boundary carry occurs.

Multiplication. A multiplication example is a pair of decimal integers (A, B), and the target is

$$f^\star_{\text{mult}}(A, B) := A \cdot B.$$

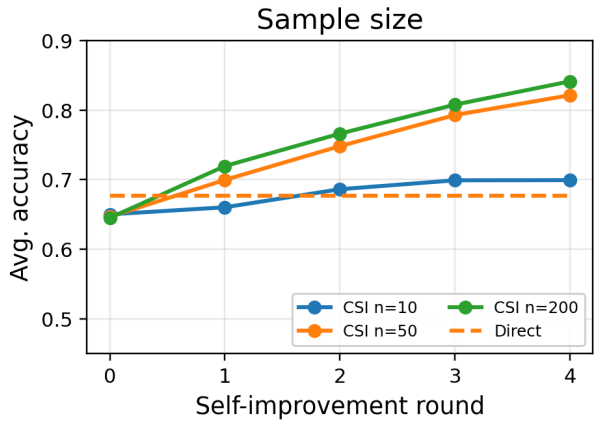
For an n-digit by m-digit problem, the composition rule follows the schoolbook decomposition. If $m \leq n$, write

$$B = \sum_{j=0}^{m-1} b_j 10^j,$$

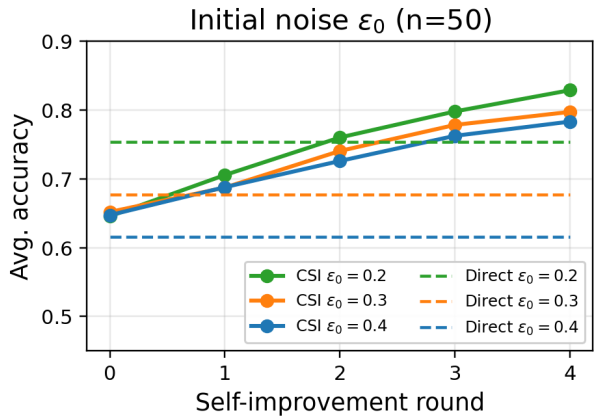
where each b_j is a single decimal digit. The product is then

$$A \cdot B = \sum_{j=0}^{m-1} (A \cdot b_j) 10^j.$$

Thus the pseudo-label for a larger multiplication can be constructed from predictions on one-digit-by-n component products, shifted by place value, and summed. When $n < m$, we symmetrically decompose A instead. In the experiment in Section 3.1, the aggregation sum is performed by the addition model produced by the addition self-improvement procedure, and each stage introduces new operand shapes (n, m) and (m, n); the main heatmap averages held-out accuracy across shapes with the same total operand size $n + m$.



(a) Effect of composed sample size per round.



(b) Effect of seed noise.

Figure 5. Synthetic concatenation experiments. CSI denotes compositional self-improvement. Increasing the number of composed examples per round improves recovery beyond direct composition, while higher seed noise degrades performance and slows recovery.

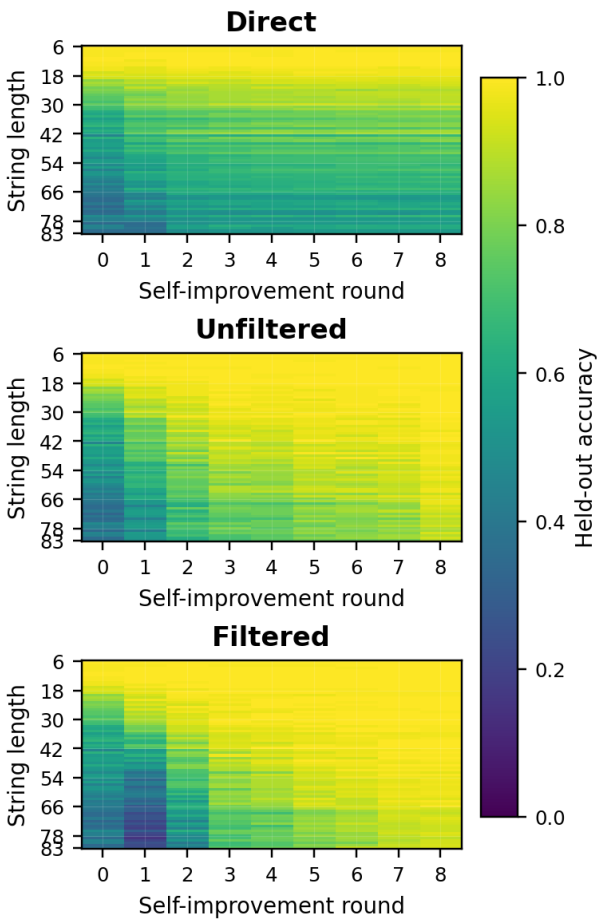


Figure 6. Run-length baseline heatmaps on the ten-symbol task with symbol-and-length outputs. Direct pseudo-labeling lags behind both composition variants; filtered composition reaches the largest reliably solved lengths, while unfiltered composition remains highly competitive, unlike in addition.

C.2. Synthetic concatenation experiment

We use a controlled synthetic task to illustrate the effects of the two quantities highlighted by the theory: composed sample size and pseudo-label noise.

Setup. Inputs are bit strings $x \in \{-1, +1\}^K$ with sum $s(x) = \sum_i x_i$ and label $\mathbf{1}\{s(x) \geq 0\}$. Composition concatenates strings, so sums compose additively: $s(x \circ x') = s(x) + s(x')$. We simulate a seed predictor by independently corrupting primitive bits with error rate ϵ_0. *Direct composition* applies this noisy predictor without retraining. Compositional self-improvement constructs larger examples round by round, pseudo-labels each composed example by summing the predicted component sums, retains a fixed number of composed examples per round, and trains a ridge regression model to predict the sum.

Results. Figure 5 supports the theoretical predictions. With too few retained composed examples, CSI remains close to direct composition; with enough examples, it steadily improves across rounds. Increasing the primitive error rate weakens the signal and slows recovery. This illustrates the two factors emphasized by the theory: the amount of composed data and the noise level of the seed predictor.

C.3. Run-Length Baseline Comparisons

We compare the same three baselines as in the addition study: *direct pseudo-labeling*, *unfiltered composition*, and *filtered composition*.

Setup. For run-length, *direct pseudo-labeling* labels larger examples in a single step, while *unfiltered composition* applies the composition rule without a safety filter. *Filtered composition* uses the same rule but removes examples whose

composed labels are unreliable due to boundary-continuing runs.

Results. Figure 6 shows that composition drives the run-length gain. Direct pseudo-labeling reaches 0.774 final accuracy and exceeds 0.9 accuracy only up to length 30. Unfiltered and filtered composition reach $0.972/0.978$ final accuracy and maintain > 0.9 accuracy through lengths $82/83$, respectively, so filtering adds only a small gain beyond composition. One reason may be that boundary continuations that change the answer are relatively rare under the natural evaluation distribution. We therefore perform a more controlled boundary evaluation next.

C.4. Run-Length Boundary Diagnostics

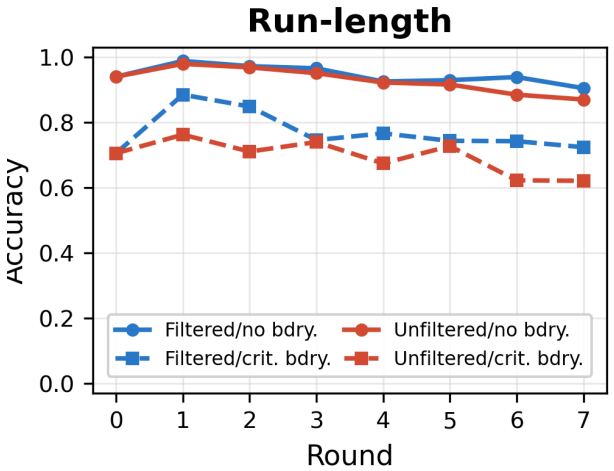
We test whether the learned model generalizes to boundary-continuation cases excluded by the safety filter. These diagnostics are designed to distinguish ordinary generalization to filtered examples from failures caused by limited output coverage in the composed training data.

Setup. During training, run-length examples are generated by equal-length two-block composition: a length-n string is formed by concatenating parts of lengths $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$. The safety filter rejects same-symbol boundary runs, since they may merge and change the label. Therefore, accepted composed examples only contain longest runs that are already visible within one part.

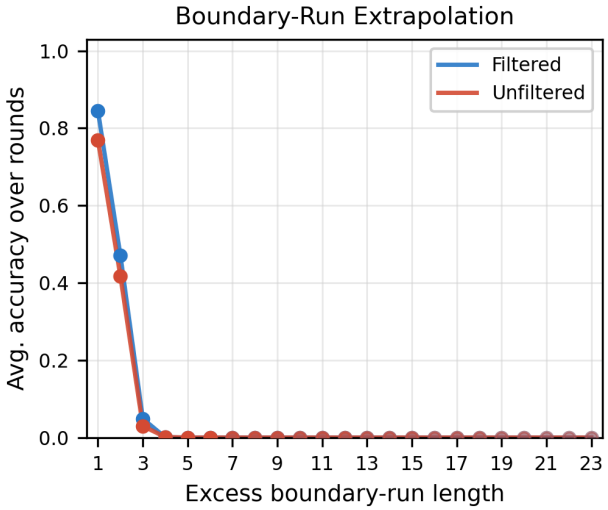
For evaluation, we construct controlled boundary-continuation slices. The critical-boundary slice consists of strings whose longest run is created by merging two same-symbol runs across the middle boundary. We then vary the merged boundary-run length beyond the maximum run length represented by accepted two-block compositions, which is bounded by the larger block length $\lceil n/2 \rceil$. When feasible, each part also contains a competing run one shorter than the boundary run.

Results. Figure 7 shows two main patterns. First, filtered composition consistently outperforms unfiltered composition on longer sequences, although the gap is smaller than in addition. This suggests that filtering helps control structured errors, but its effect is more limited for run-length.

Second, boundary-run extrapolation reveals a clear failure mode. As the merged boundary run exceeds the larger-block limit $\lceil n/2 \rceil$, performance drops sharply. This reflects an output-coverage limitation of the fixed two-block composition rule: accepted composed examples never require predicting runs longer than those already present in a component. Thus, runs longer than the component-length bound receive little or no pseudo-labeled supervision, and the model does not generalize reliably to that regime.



(a) Fixed-boundary slices.



(b) Boundary-run extrapolation.

Figure 7. Run-length boundary diagnostics. In the fixed-boundary diagnostic, the no-boundary slice has different boundary symbols, so no run can merge across the split. The critical-boundary slice has same-symbol boundary runs whose merger is longer than any run within either half, making the cross-boundary run determine the true answer. In the boundary-run extrapolation diagnostic, the merged boundary run exceeds the maximum run length represented by the fixed two-block composition rule, i.e., the larger-block limit $\lceil n/2 \rceil$. The x-axis reports this excess length; accuracies are averaged over all rounds.

Overall, these diagnostics suggest a general design principle for compositional self-improvement: filtering should be paired with sufficient coverage of the target behavior. Clean pseudo-labels alone may not be enough if the accepted compositions systematically exclude part of the relevant output range. This motivates diversifying composition paths, for example by using multiple split positions or unbalanced splits such as lengths $n-1$ and 1, so that the composed data

covers the regimes needed for reliable generalization.

C.5. Seed Strength and Composed-Data Size

Compositional self-improvement can fail when the seed model is too unreliable or when too few accepted pseudo-labeled examples are available at each round. We vary both factors on the two main real tasks.

Setup. For run-length, the model predicts the repeated symbol and length of the longest run over a ten-symbol alphabet; composition rejects examples where a run may continue across the boundary. For addition, we use the filtered carry-aware composition rule from the main experiments. For each task, we select weak, moderate, and strong seed checkpoints by their worst accuracy on the seed range: 0.73, 0.81, and 0.99 for run-length, and 0.75, 0.84, and 0.995 for addition. We vary the number of retained composed examples per new size: 250, 500, and 2000 per length for run-length, and 2500, 5000, and 10000 per digit for addition.

Results. Figures 8 and 9 show that stronger seeds and larger composed datasets improve self-improvement. For run-length, increasing retained examples from 250 to 2000 per length substantially expands reliable performance across seed strengths. For addition, larger composed pools also help, especially with a strong seed; the 10000 example-per-digit setting gives the broadest reliable range. These results support the prediction that self-improvement depends on both the quality of the seed model and the amount of composed data.

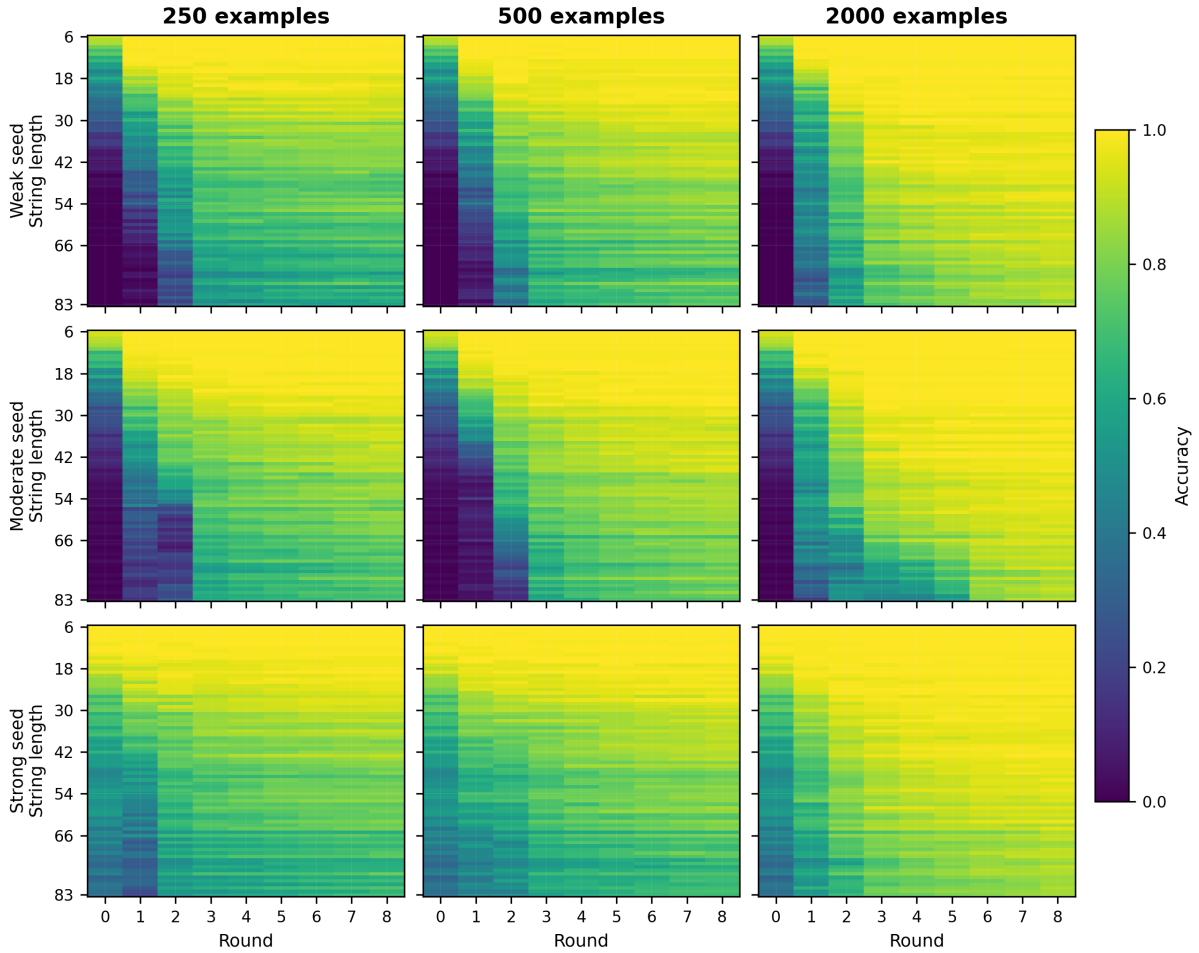


Figure 8. Run-length ablation over seed strength and retained composed-data size. Each panel shows accuracy by string length across self-improvement rounds.

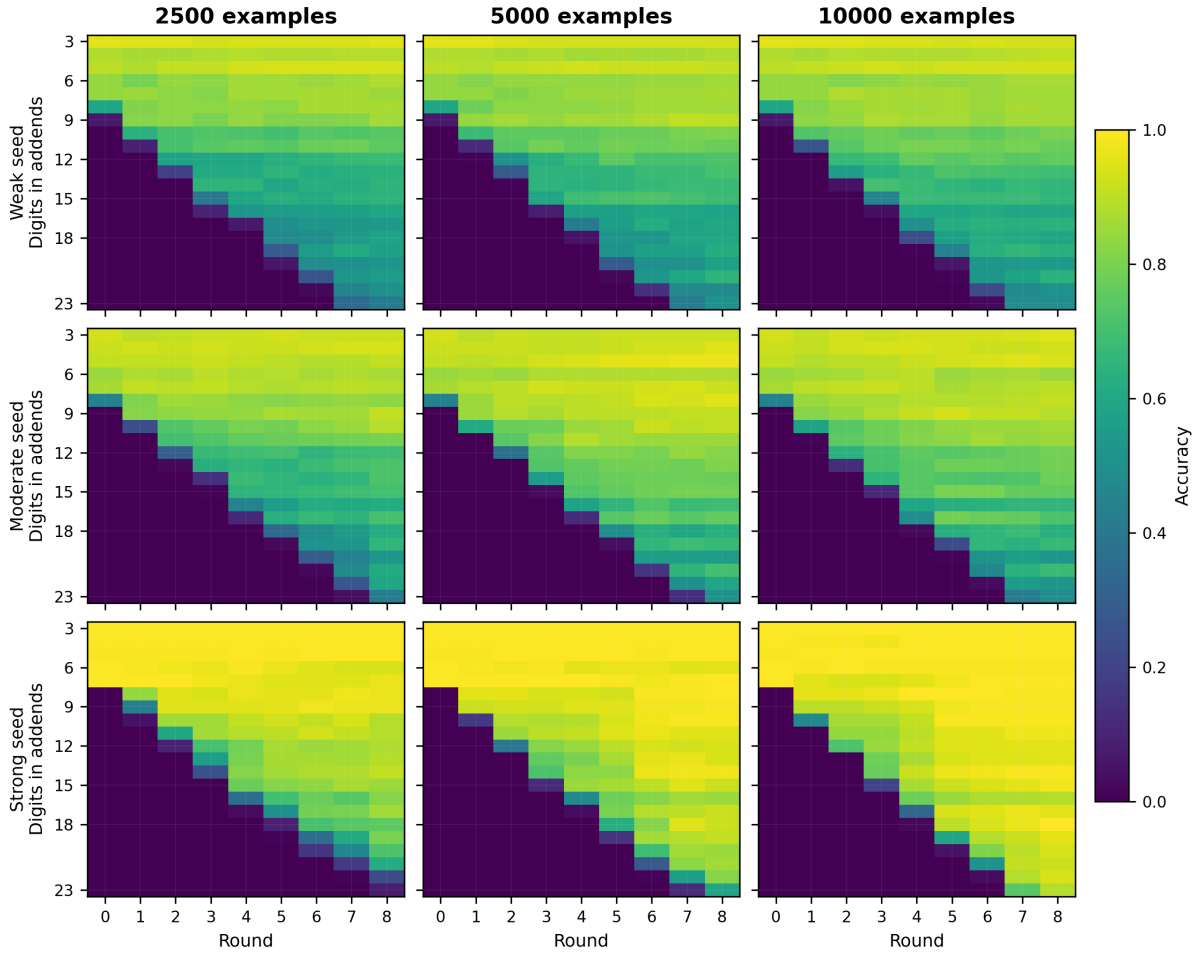


Figure 9. Addition ablation over seed strength and retained composed-data size. Each panel shows accuracy by addend digit length across self-improvement rounds.