N-Gram Induction Heads for In-Context RL: Improving Stability and Reducing Data Needs

Ilya Zisman *
AIRI
Skoltech

Alexander Nikulin
AIRI
MIPT

Andrei Polubarov
AIRI
Skoltech

Nikita Lyubaykin
AIRI
Innopolis University

Vladislav Kurenkov
AIRI
Innopolis University

Abstract

In-context learning allows models like transformers to adapt to new tasks from a few examples without updating their weights, a desirable trait for reinforcement learning (RL). However, existing in-context RL methods, such as Algorithm Distillation (AD), demand large, carefully curated datasets and can be unstable and costly to train due to the transient nature of in-context learning abilities. In this work we integrated the n-gram induction heads into transformers for in-context RL. By incorporating these n-gram attention patterns, we significantly reduced the data required for generalization — **up to 27 times fewer** transitions in the Key-to-Door environment — and eased the training process by making models less sensitive to hyperparameters. Our approach not only matches but often surpasses the performance of AD, demonstrating the potential of n-gram induction heads to enhance the efficiency of in-context RL.

1 Introduction

In-context learning is a powerful ability of autoregressive models, such as transformers (Vaswani et al., 2023) or state-space models (Gu et al., 2022), allowing them to infer and solve tasks from just a few examples without updating models weights (Brown et al., 2020). Such an adaptation ability is useful in Reinforcement Learning (RL), where after extensive pre-training, the agent can adapt on-the-fly to unseen tasks or environments (Grigsby et al., 2023; Ramrakhya et al.).

In-context Reinforcement Learning (ICRL) methods that learn from offline datasets were first introduced by Laskin et al. (2022) and Lee et al. (2023). In the former work, Algorithm Distillation (AD), authors propose to distill the policy improvement operator from a collection of learning histories of RL algorithms, after which an agents is able to generalize to

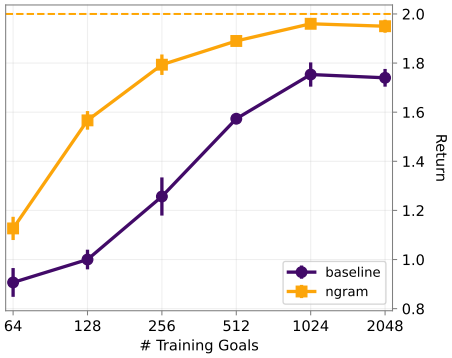

Figure 1: Performance comparison for different number of training goals between our method and Algorithm Distillation (AD), an in-context reinforcement learning method (Laskin et al., 2022). Our method demonstrates similar performance with less training goals (128 vs. 512) and in general outperforms the baseline in Key-to-Door environment. In such stricter data scenario it is able to perform on par with AD using **27x** less transitions in total. See Section 5 for details.

*Correspondence to zisman@airi.net
Work is done at dunnolab.ai

Workshop on Adaptive Foundation Models at 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

unseen tasks in-context. In the latter, authors similarly show it is possible to generalize from a dataset of interactions, provided that the optimal actions are also available.

Both methods require specifically curated data, which is demanding to obtain (Nikulin et al., 2024b). Several methods were proposed to tackle this problem (Zisman et al., 2024; Kirsch et al., 2023) by easing the data acquisition process. In addition, the in-context ability itself is transient (Singh et al., 2024) and hard to predict its emergence from the cross-entropy loss alone (Agarwal et al., 2024), making the training of such models unstable and expensive in terms of training budget. Our work aims to solve aforementioned obstacles and presents changes made to transformer's attention heads, which can significantly speed up training process and decrease the total amount of data needed for in-context learning to emerge.

It has been shown that a central mechanism that enables in-context learning in transformers is induction heads (Olsson et al., 2022). Edelman et al. (2024) studied the emergence of these statistical induction heads on synthetic data and concluded that transformers obtain a simplicity bias towards plain uni-grams. Akyürek et al. (2024) take a step forward in this direction, showing that during in-context learning of transformers, there appear higher-order induction heads in the attention mechanism, which capture different n-grams in the sequence. They propose to hardcode this mechanism into a transformer, creating an n-gram layer which is used interchangeably with standard multi-head attention mechanism. Intuitively, a transformer benefits from it by not learning this complicated behaviour by itself, rather it straightforwardly receives an inductive bias n-gram heads provide. This approach significantly boosts perplexity even when applied to recurrent sequential models, indicating that n-grams are a central mechanism for in-context learning.

We bring up these findings to ICRL setting and show that:

- **N-grams heads decrease the amount of data needed for generalization on novel tasks.** By leveraging them, it is possible to reduce the total amount of transitions in training data by **27x** compared to the original method of Laskin et al. (2022). The results are presented in Figure 3.
- **N-grams help to ease training of in-context models.** By employing n-gram heads, one needs considerable less time doing hyperparameters search, thus making a model less sensitive to hyperparameters and making it cheaper to train. The results are presented in Figure 2.

2 Related Work

In-context RL. The key feature behind ICRL is the adaptation ability of a pretrained agent . In general, it relies on the transformer's ability to infer a task from the history of interactions with an environment. Müller et al. (2021) show that transformers are capable of performing Bayesian inference, which is known for its applicability for reasoning under uncertainty (Ghavamzadeh et al., 2015). Laskin et al. (2022) proposed to pretrain a transformer on learning histories of RL algorithms which allows it to implicitly learn policy improvement operator. During inference on unseen tasks, a transformer is able to improve its policy by observing a context and to infer a task from it. However, such approach requires specific datasets, which may be expensive to collect (Nikulin et al., 2024b). To tackle this, it has been proposed to generate datasets following noise curriculum instead of training thousands of agents (Zisman et al., 2024) or make augmentations to existing data (Kirsch et al., 2023). Our work follows the direction of democratizing data restrictions, but instead of working with data, we introduce a model-centric approach, making a transformer to perform in-context reinforcement learning with less data.

N-Gram and Transformers. N-Gram statistical models has been known for decades and used in the statistical approach to language modelling (Brown et al., 1992; Kneser & Ney, 1995). More recent approaches (Roy et al., 2022; Liu et al., 2024) study the application of n-grams to transformer models, finding that they can increase the overall performance. Akyürek et al. (2024) discover that a transformer implicitly implements 2-gram attention pattern when solving in-context learning task, which authors denote as a higher order of induction head (Olsson et al., 2022). They explicitly implement 1-, 2- and 3-gram attention layers and observe a significant reduction in perplexity. Another work (Edelman et al., 2024) directly investigates the behavior of n-gram induction heads during training process. Authors find that transformers are biased towards simple solutions, thus

making it problematic for higher order induction heads to appear. To our knowledge, we are the first to apply these findings to decision making settings.

3 Method

We build our method on AD (Laskin et al., 2022) as our baseline. In its core, it uses learning histories of RL algorithms that are trained to solve a single task in an environment. The data with learning progress of multiple RL agents then passed to a transformer which learns to predict the next action via cross-entropy loss. This results in a transformer that is able to adapt to unseen tasks on inference without any weight updates. The details of implementation can be found in Appendix A.

However, AD suffers the same problems as any in-context algorithm does. Learning of the optimal solution can be delayed by a tendency of transformers to learn simple structures at first (Edelman et al., 2024). Besides, the nature of in-context ability is unstable and can fade into in-weights regime as the training progresses, considerably complicating the emergence of adaptation ability (Singh et al., 2024).

To combat these obstacles, we implement n-gram attention layer (Akyürek et al., 2024) as one of the layers of a transformer. In essence, it hardcodes computations of n-gram statistics into the transformer itself, rather than waiting for them to emerge naturally. The attention pattern that is calculated from the input sentence is defined as:

$$A(n)_{ij} \propto \mathbb{1}\left[(\wedge_{k=1}^{n} x_{i-k} = x_{j-k-1})\right]$$

After which, we apply a projection and add a residual to the output:

$$\texttt{NGH}^n\left(h^l\right) = W_1 h^l + W_2 A(n)^\top h^l$$

Where n is the n-grams length, W_1 and W_2 are learnable projection matrices and h^l is an embedding from a previous transformer layer. In simple terms, we look for n-gram occurrences and with the help of $A(n)$ attention pattern force gradients to flow only through tokens that co-occur in the sequence.

To count n-gram statistics we use raw input sequence. However, since we are working in RL setting, the input sequence has a form of $(s_0, a_0, r_0, ..., s_n, a_n, r_0)$, so in our experiments we tested two approaches. We either compare the equivalence of full transitions $(s_i, a_i, r_i) = (s_j, a_j, r_j)$ or just states $(s_i = s_j)$.

Throughout the text we use the terms *learning histories* and *tasks*. The task is a predefined grid or a pair of grids an agent must come to upon it receives a reward. The learning history is an ordered collection of states, actions and rewards an RL algorithm observed (or produced) while learning to solve a *single* task. When we say we generated a dataset of n tasks with m learning histories, it means for each of the task there are at least $\lfloor \frac{m}{n} \rfloor$ learning histories per task. Unlike Laskin et al. (2022), we distinguish between tasks and learning histories, as it is often the case with real data when many trajectories correspond to only a few tasks (Yu et al., 2019; Gallouédec et al., 2024).

4 Experiment Setup

We test our method on two environments, Dark Room and Dark Key-to-Door, originally presented by Laskin et al. (2022). Both environments are grid-worlds of size 9x9. In Dark Room, an agent is spawned at the center and needs to find a goal, after which it receives a reward of 1. The task for Key-to-Door is similar, but at first an agents seeks for the key which allows it to open the door (both of these actions lead to getting a reward of 1). For each environment, we access the performance of ICRL algorithms only on *unseen* tasks. The total number of goals is 81 and 6561 for Dark Room and Key-to-Door respectively. The more detailed description can be found in Appendix C.

To show that our method is more stable, we choose to report the results using the Expected Max Performance protocol (EMP) (Dodge et al., 2019; Kurenkov & Kolesnikov, 2022). By doing so, we do not report the maximum performance of a single checkpoint, rather we show the expected performance for a certain computational budget. By using this approach we simultaneously compare our method with a baseline in terms of easiness of training and maximum achieved performance.

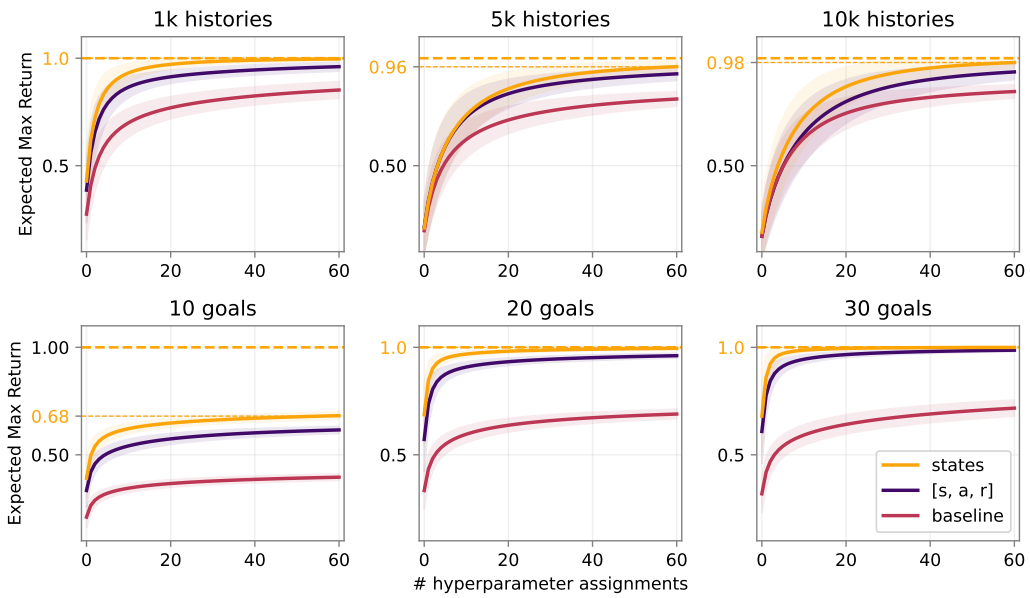

Figure 2: Results on Dark Room. We search through hyperparameters in random order and report expected maximum performance (Dodge et al., 2019). We also constrain learning time of tested algorithms to find the hyperparameters that ensure faster convergence. **The top row** shows experiments with different number of learning histories, with the total number of training goals fixed. It is seen that our method needs much less hyperparameter assignments (20 for 1K histories) to find the optimal model, while the baseline performance increases only asymptotically (full plots are shown in Appendix E). **The bottom row** presents experiments with varied number of goals and fixed number of learning histories. Our method makes it possible to find the optimal hyperparameters with only 15 hyperparameter assignments, while the baseline fails to work in such low data conditions. However, none of the methods can learn to generalize from only 10 goals.

For each experiment we make 500 and 400 hyperparameter assignments in total for Dark Room and Key-to-Door respectively. To further demonstrate the effectiveness of our method, we constrain the optimization by 10K gradient steps to find which algorithm is faster to converge to in-context learning regime. The exact hyperparameter assignments setups are shown in Appendix D.

5 Results

We start with showing the ability of our method to significantly reduce the hyperparameter sensitivity of AD. In the top row of Figure 2 we vary the number of learning histories, when the number of training goals is fixed. It can be seen that for our method to converge to the optimal hyperparameters, it needs only about 20 hyperparameter assignments. At the same time, for a baseline AD one needs to search through 400 different hyperparamenter combinations on average to find a model that can solve this relatively small environment.

In the next set of experiments we investigate an ability of our method to work in low-data regime. We fix the number of learning histories and vary the number of goals only. Note that in the previous experiment we had 60 tasks for training, but now we considerably reduce their quantity. In the bottom row of Figure 2 the experiments with $10, 20, 30$ training goals are shown. Neither method can generalize to unseen task when presented only with 10 training goals. However, it is different for 20 and 30 goals, where our method finds the optimal model in approximately 15 hyperparameter assignments. Noticeably, the baseline method completely fails to learn from the constrained data, which highlights the applicability of our method when no large dataset is available. For transparency reasons, we show the full-length plots in Appendix E.

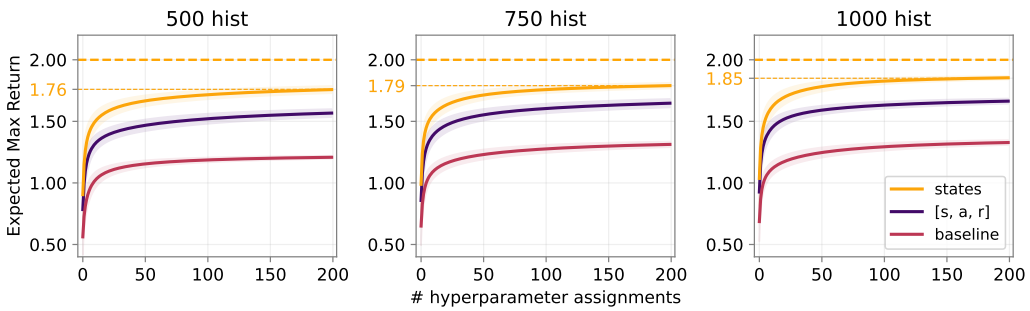

Figure 3: Results on Key-to-Door. We demonstrate the ability of our method to generalize when the data is extremely low in a more complex environment than Dark Room. We fix the total number of goals with 100, significantly shrinking the number of learning histories. Keep in mind that for the baseline method to converge to a model with the same return, it needs 2048 goals and 2048 learning histories (Laskin et al., 2022). We show that our method needs **27x** data. The baseline method can no longer converge with that few data and its performance plateaus with the increasing number of hyperparameter assignments.

Next, we further restrict the amount and diversity of data available to train. We set up an experiment in Key-to-Door, a more comprehensive environment with the total of 6561 tasks, with only 100 training tasks and $500, 750, 1000$ learning histories. Comparing to Laskin et al. (2022), we use **27x** less data. The detailed calculations are provided in Appendix B. It can be observed from Figure 3 that the baseline method cannot find a model that is able to generalize to unseen goals in such a setting. In turn, out method demonstrates performance on par with what Laskin et al. (2022) report in their work. To ensure that our implementation of a baseline (AD) can solve the environments, we present the performance of a baseline that is trained on optimal hyperparameters in Appendix F.

6 Conclusion and Future Work

In our work we show that incorporating n-gram induction heads can significantly ease training of in-context reinforcement learning algorithms. Our findings are twofold: (**i**) we show that n-gram heads can notably decrease a sensitivity to hyperparameters of in-context RL algorithm and (**ii**) we demonstrate that our method is able to generalize from much less data than the baseline Algorithm Distillation (Laskin et al., 2022) approach. We speculate these findings are mainly attributed to the imperfect nature of in-context learning itself: a tendency of transformers to converge to simple solutions first (Edelman et al., 2024) and transitivity of in-context ability itself (Singh et al., 2024).

While we believe our findings are promising, there are some limitations of current work. Further research is needed to make our method compatible with continuous observations, which can greatly expand the applicability of the method. Also, one might consider scaling to larger models and more comprehensive environments, e.g. XLand-Minigrid (Nikulin et al., 2024a) or Meta-World (Yu et al., 2019), which are not yet solved.

References

Agarwal, R., Singh, A., Zhang, L. M., Bohnet, B., Chan, S., Anand, A., Abbas, Z., Nova, A., Co-Reyes, J. D., Chu, E., et al. Many-shot in-context learning. *arXiv preprint arXiv:2404.11018*, 2024.

Akyürek, E., Wang, B., Kim, Y., and Andreas, J. In-context language learning: Arhitectures and algorithms. *arXiv preprint arXiv:2401.12973*, 2024.

Brown, P. F., Della Pietra, V. J., Desouza, P. V., Lai, J. C., and Mercer, R. L. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480, 1992.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R.,

Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling, 2021. URL https://arxiv.org/abs/2106.01345.

Dodge, J., Gururangan, S., Card, D., Schwartz, R., and Smith, N. A. Show your work: Improved reporting of experimental results. *arXiv preprint arXiv:1909.03004*, 2019.

Edelman, B. L., Edelman, E., Goel, S., Malach, E., and Tsilivis, N. The evolution of statistical induction heads: In-context learning markov chains. *arXiv preprint arXiv:2402.11004*, 2024.

Gallouédec, Q., Beeching, E., Romac, C., and Dellandréa, E. Jack of all trades, master of some, a multi-purpose transformer agent, 2024. URL https://arxiv.org/abs/2402.09844.

Ghavamzadeh, M., Mannor, S., Pineau, J., Tamar, A., et al. Bayesian reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 8(5-6):359–483, 2015.

Grigsby, J., Fan, L., and Zhu, Y. Amago: Scalable in-context reinforcement learning for adaptive agents. *arXiv preprint arXiv:2310.09971*, 2023.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces, 2022. URL https://arxiv.org/abs/2111.00396.

Kirsch, L., Harrison, J., Freeman, D., Sohl-Dickstein, J., and Schmidhuber, J. Towards general-purpose in-context learning agents. Workshop on Distribution Shifts, 37th Conference on Neural Information . . . , 2023.

Kneser, R. and Ney, H. Improved backing-off for m-gram language modeling. In *1995 International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pp. 181–184 vol.1, 1995. doi: 10.1109/ICASSP.1995.479394.

Kurenkov, V. and Kolesnikov, S. Showing your offline reinforcement learning work: Online evaluation budget matters. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11729–11752. PMLR, 17–23 Jul 2022.

Laskin, M., Wang, L., Oh, J., Parisotto, E., Spencer, S., Steigerwald, R., Strouse, D., Hansen, S., Filos, A., Brooks, E., et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.

Lee, J. N., Xie, A., Pacchiano, A., Chandak, Y., Finn, C., Nachum, O., and Brunskill, E. Supervised pretraining can learn in-context reinforcement learning. *arXiv preprint arXiv:2306.14892*, 2023.

Liu, J., Min, S., Zettlemoyer, L., Choi, Y., and Hajishirzi, H. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*, 2024.

Müller, S., Hollmann, N., Arango, S. P., Grabocka, J., and Hutter, F. Transformers can do bayesian inference. *arXiv preprint arXiv:2112.10510*, 2021.

Nikulin, A., Kurenkov, V., Zisman, I., Agarkov, A. S., Sinii, V., and Kolesnikov, S. Xland-minigrid: Scalable meta-reinforcement learning environments in jax. In *Automated Reinforcement Learning: Exploring Meta-Learning, AutoML, and LLMs*, 2024a.

Nikulin, A., Zisman, I., Zemtsov, A., Sinii, V., Kurenkov, V., and Kolesnikov, S. Xland-100b: A large-scale multi-task dataset for in-context reinforcement learning. *arXiv preprint arXiv:2406.08973*, 2024b.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Ramrakhya, A. E. G. C. R., Yadav, K., and Szot, D. B. Z. K. A. Relic: A recipe for 64k steps in-context reinforcement learning for embodied ai.

Roy, A., Anil, R., Lai, G., Lee, B., Zhao, J., Zhang, S., Wang, S., Zhang, Y., Wu, S., Swavely, R., et al. N-grammer: Augmenting transformers with latent n-grams. *arXiv preprint arXiv:2207.06366*, 2022.

Singh, A., Chan, S., Moskovitz, T., Grant, E., Saxe, A., and Hill, F. The transient nature of emergent in-context learning in transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

Sinii, V., Nikulin, A., Kurenkov, V., Zisman, I., and Kolesnikov, S. In-context reinforcement learning for variable action spaces. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 45773–45793. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/sinii24a.html.

Tarasov, D., Nikulin, A., Akimov, D., Kurenkov, V., and Kolesnikov, S. Corl: Research-oriented deep offline reinforcement learning library. *Advances in Neural Information Processing Systems*, 36, 2024.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1910.10897.

Zisman, I., Kurenkov, V., Nikulin, A., Sinii, V., and Kolesnikov, S. Emergence of in-context reinforcement learning from noise distillation. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

A Model Implementation

We build our model on the base of Decision Transformer (Chen et al., 2021), which implementation is taken from CORL (Tarasov et al., 2024). We modify it by removing return-to-go completely. Since in RL we operate with tuples of states, actions and rewards, we concatenate them into one large "token" to preserve sequence length size as in (Lee et al., 2023; Sinii et al., 2024).

B Calculation of Transitions in Data

In appendix I of Laskin et al. (2022) they mention that AD is more data-effective than source algorithm and report the size of a dataset. The total number of data needed to achieve an approximate of 1.81 return on Key-to-Door [2] is reported as

> (...) on 2048 Dark Key-to-Door tasks for 2000 episodes each.

The estimate of total number of transitions *to generate* for AD, considering the maximum length of an episode in Key-to-Door is 50 steps, equals: $2048 \times 2000 \times 50 = 204.8\texttt{M}$ transitions.

We generate 100 unique training tasks and then sample 750 train task with repetition from the original 100. Then we make 200 training episodes for each task. In total, we get $750 \times 200 \times 50 = 7.5\texttt{M}$ transitions, which is more than **27x** less data.

C Environments

Dark Room. 2D POMDP with discrete state and action spaces (Laskin et al., 2022). The grid size is 9×9, where an agent has 5 possible actions: up, down, left, right and do nothing. The goal is to find a target cell, the location of which is not known to the agent in advance. The episode length is fixed at 20 time steps, after which the agent is reset to the middle of the grid. The reward $r = 1$ is given for every time step the agent is on the goal grid, otherwise $r = 0$. The agent does not know the position of the goal, hence it is driven to explore the grid. In total, there are 81 goals.

Key-to-Door. Similar to *Dark Room*, but it first requires an agent to find an invisible key and then the door. Without a key, the door will not open. The reward is given when the key is found ($r = 1$) and once the door is opened (also $r = 1$), after which the game terminates. The agent then resets to a random grid. The maximum episode length is 40, and since we can control the location of the key and door, there are around 6.5k possible tasks.

[2] since no accurate data of plots was published, we used free-to-use WebPlotDigitizer for Fig. 6 in AD paper

D Sweeps Setup

We use weights and biases sweep for running sweeps. All of the sweep setups are available by this clickable link [will be available for camera-ready version].

We also report the setup of hyperparameter sweep in the table below.

Table 1: Hyperparameter search space

	Distribution	Values
seq len	-	[100, 150, 200, 250]
pre norm	-	[true, false]
normalize qk	-	[true, false]
label smoothing	uniform	[0.0, 0.8]
learning rate	log uniform	[1e-4, 1e-2]
weight decay	log uniform	[1e-7, 2e-2]
residual dropout	uniform	[0.0, 0.5]
embedding dropout	uniform	[0.0, 0.9]
episode subsample	-	[1, 2, 4, 8, 20]

E Full Plots

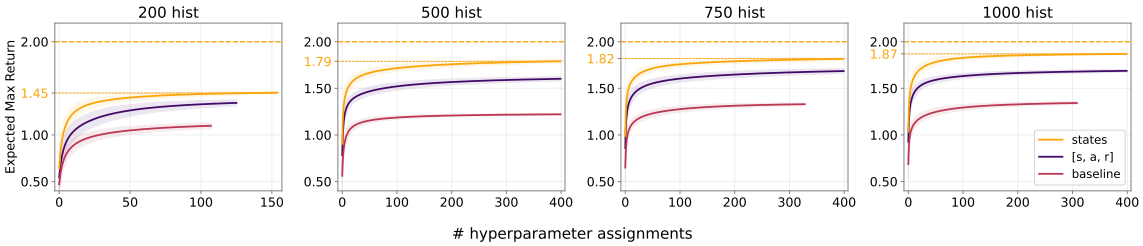

Figure 4: Full length plots for Key-to-Door. For 200 learning histories we halted the sweep early, since it was obvious the performance has stalled.

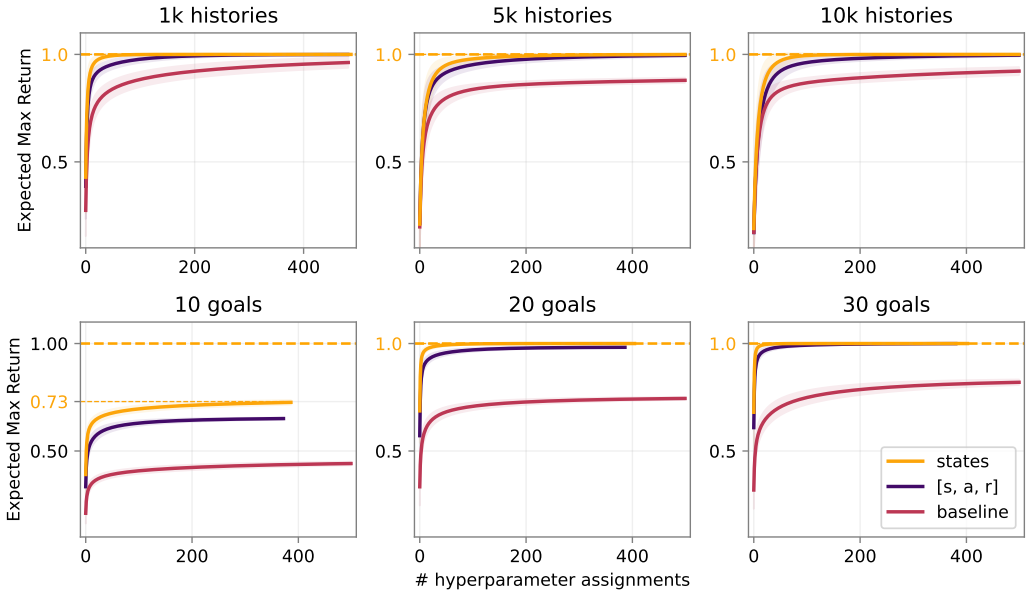

Figure 5: Full length plots for Dark Room. Some of the computations halted earlier for the same reason as in Figure 4

F Performance of AD on Key-to-Door and Dark Room

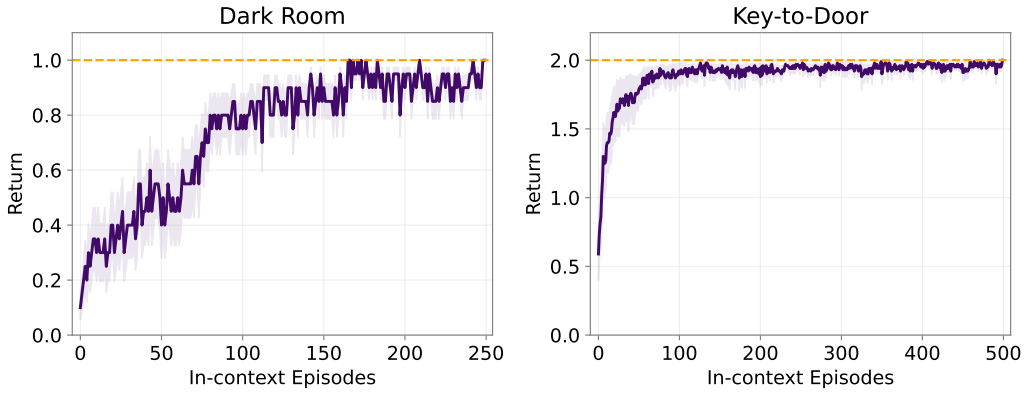

Figure 6: AD performance on Dark Room and Key-to-Door. This plot shows that our implementation of AD demonstrates optimal performance given the right hyperparameters.