APPROXIMATING LONGER BIRTH AND DEATH PROCESSES WITH LARGE LANGUAGE MODELS AND THE SELF IMPROVEMENT FRAMEWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Birth and Death Processes are continuous-time Markov chains that transition to either the next or previous state. They model population sizes; state transitions lead to the births and deaths of individuals in a population. We model large,structured birth and death processes on large language models by simulating population data and training large language models to predict the next population. As large language models struggle with large and complicated inputs, we apply previously studied length generalization methods to the problem, including extending the self improvement framework to data from birth and death processes to benchmark performance. To overcome observed underfitting caused by low difficulty added per self improvement framework round, we parameterize the variance added by each self improvement round through the number of population states added. The Chapman-Kolmogorov equation shows us that the number of population states added each self improvement round influences the variance that the large language models fit to. Finally, we compare applied methods of length generalization in this domain. Our source code can be found on Github at: `https://anonymous.4open.science/r/self_improving_bd_processes/README.md`.

1 INTRODUCTION

A Birth and Death Process is formally defined as a continuous-time Markov chain with states $\{i \in \mathbb{Z} \mid 0 \leq i \leq n\}$ for which transitions from state i go to either state i - 1 or state i + 1 Ross (1995). The state represents the population. Individuals die when a transition to state i - 1 occurs. Individuals are born when a transition to state i + 1 occurs. We define the rate at which an individual is born as λ_i, and the rate at which they die as μ_i. If birth and death rates are parameterized, we name them as $\lambda_i(\theta)$ and $\mu_i(\theta)$. A transition to state i + 1 occurs with a probability of $\frac{\lambda_i(\theta)}{\mu_i(\theta)+\lambda_i(\theta)}$ and a transition to state i - 1 occurs with a probability of $\frac{\mu_i(\theta)}{\mu_i(\theta)+\lambda_i(\theta)}$. As a continuous time markov chain, a Birth and death process defined as $Z \equiv \{Z(t), t \geq 0\}$ tracks the stochastically evolving number of individuals in a population over time.

Evolutionary biology Novozhilov et al. (2006), ecology Crawford & Suchard (2012), population genetics, and epidemiology Doss et al. (2013) include applications that really heavily on Birth and Death Processes Ho et al. (2018). Many real world population sizes can be predicted through accurate transition matrices. They inherently track only one population. Monomolecular reaction systems and the stochastic susceptible-infected-recovered (SIR) model can both be represented by Birth and Death Processes. The potential of large language models to learn these niche patterns therefore yields benefits in these applications; we can train a large language model in these situations with approximate results without access to real world data. We seek to approximate these patterns with large language models and predict results beyond the initial training distribution.

Large language models struggle to handle complex problem instances beyond their initial training distribution Anil et al. (2022). While handling a regular sized Birth and Death Process is not a trivial task, handling larger data has proven difficult, especially after initial training. Prior studies have shown length generalization remains a difficult problem for most large language models.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

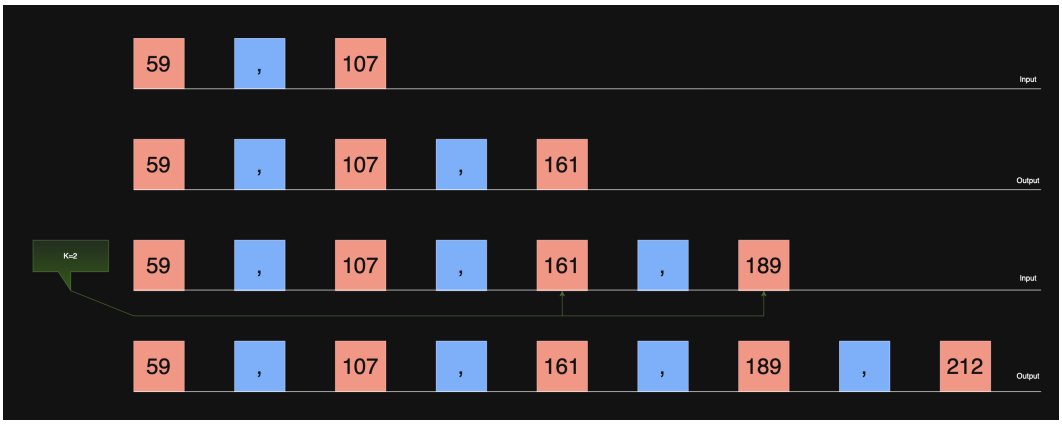


Figure 1: A Birth and Death Process, as input into a self-improvement trained language model. The input consists of a sequence containing two population sizes and separator tokens in between; part of the task is for the model to understand the structure and to put separator tokens in between. In this image, we show what might happen with the parameter $K = 2$; everytime we increase the length of input x_i during self improvement training, we add two population sizes to the input chain, instead of incrementing by one at a time. The Chapman-Kolmogorov equation shows us that K influences the variance of our predicted population sizes.

The self improvement framework, as introduced in 2025, involves training large language models on longer observations without access to hard labels; it incrementally replaces ground truth with the results of the previous model before more training Lee et al. (2025). This approach has shown that transformers can gradually generalize on some problems, such as the reverse string task.

Does the self improvement framework extend to Birth and Death Processes? Does it, or other length generalization solutions, allow us to tackle larger Birth and Death Processes as well as the structure they are represented with? How do the specific properties of Birth and Death Properties allow us to apply the Self Improvement Framework? In this paper, we investigate how to approximate very large Birth and Death Processes with large language models.

2 Prior work

Length generalization of large language models has been studied in the past Anil et al. (2022). While large language models are outstanding at numerous pattern recognition tasks, length generalization in general remains an unsolved problem. Extrapolating from short problem instances to out of distribution examples is important when dealing with complex examples. Length generalization in large language models, in general, faces difficulty.

The ability of a transformer, the basis of a language model, to extend any particular task varies Zhou et al. (2024). The RASP-L programming language, a specialized language that maps high level instruction codes to the internal operations of a transformer, trains transformers that yield almost perfect accuracy on count, addition, and other tasks with an additional length of up to 70 over the max training length. Some transformers perform length generalization well for some tasks; this can happen for simple reasons. Other tasks are difficult. Forward Addition and Parity are two tasks that struggle with length generalization.

Existing methods of length generalization succeed with different qualities. Fine-tuning, prompting, and scratch-pad are three methods of teaching a large language model to learn from shorter instances and handle longer ones Anil et al. (2022). Prior studies show that generalizing in length is a challenge for language models up to the 100B parameter scale Anil et al. (2022). Fine-tuning also yields mixed results.

The self improvement framework approach enables models to iteratively generate and learn from their own solutions Lee et al. (2025). Models trained on the self improvement framework can tackle harder problems with a standard transformer architecture. It extends the capabilities of a model

to longer lengths with more rounds. Self-improvement training enables transformers to generalize from easy to hard problems without labels.

2.1 The Self Improvement Framework

The Self Improvement Framework trains models by starting with a basic dataset $\mathcal{D}_0$ and iteratively increasing the difficulty of the task, which starts at d_r. In a Birth and Death Process, we increase the difficulty by increasing the number of population states in sample x_i. It then uses results of the model from the previous round to train the next round. Our datasets $\mathcal{D}_r$ are defined by the equation below Lee et al. (2025).

$$\mathcal{D}_r = \{(x_i, M_{r-1}(x_i))\}_{i=1}^{N}, \quad d_{r-1} \leq \text{Difficulty}(x_i) \leq d_r$$

At self improvement round r, we train a model M_r on all datasets $\{\mathcal{D}_i\}_{i=0}^{r}$. Self-improvement trained models can solve problems beyond their initial training distribution.

2.2 BirDePy

We simulate population data through the Python BirDePy package, a package for modeling Population Size Dependent Birth and Death Processes that expands availability to biologists and other practicioners Hautphenne & Patch (2024). The BirDePy package offers models with different parameters that control birth and death rates. The Verhulst Model, a population growth model, can be implemented as a Birth and Death Process. The BirDePy implementation of the Verhulst model parameterizes the birth and death rates $\lambda_i(\theta)$ and $\mu_i(\theta)$ with the growth and recovery rates $\theta = \{\beta, \nu\}$. A Poisson distribution may be useful in reflecting the average number of births from a continuous-time markov chain in a period of time Thompson (2001). The Ricker model, often used in population ecology, models density-dependent population growth Ricker (1973). These models can all be simulated by BirDePy.

2.3 Verhulst Model

The Verhulst Model is also known as the logistic growth model. It was introduced by Pierre Francois Verhulst in 1838 Bacaër (2011). It simulates population growth that approaches a carrying capacity. We define the growth per unit of time. We grow the population at a rate of $\frac{dP}{dt} = rP(1 - \frac{P}{K})$.

2.4 Poisson Process

The Poisson process is a probability distribution that gives the average number of events occurring randomly over a continuous period of time at a constant rate of λ Thompson (2001). We define the probability of exactly K events occurring, using $P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$.

2.5 Ricker Model

The Ricker Model is a population dynamics model that predicts future events. It approximates many situations in fish and fishery biology. We approximate using $P(X_{t+1} = k|X_t) = \frac{\lambda^k e^{-\lambda}}{k!}, \lambda = X_t e^{r(1-\frac{X_t}{K})}$.

3 Methodology

We train large language models on data from Birth and Death Process simulations to approximate them. To enable the language models to learn these transition probabilities and structural relationships, we train on large datasets before applying length generalization techniques.

We represent Birth and Death Processes through string representations before feeding them to a large language model. We tokenize the input by word, so that each population size and separator token has its own token. The population sizes are separated by separator tokens so that the large language models may function recursively, outputting string representations of the next time interval.

In other words, we obtain all $\{Z(s) : 0 \leq s < t+1\}$ from a language model with the input $\{Z(s) : 0 \leq s < t\}$; we may recurse until an absorption state is reached.

3.1 MEASUREMENTS

Assessing the quality of the approximation of a Birth and Death Process requires metrics that measure modeling quality and structure. Accuracy and average length difference measure structure, while mean absolute percentage error and mean squared error measure modeling quality.

- **Accuracy** - The percentage of tokens output by the large language model that match our ground truth approximation.
- **Length Difference** - Measures that the structure output by the large language model is correct, ex, an input of "23,40,56" returns one with the same beginning as well as a population sized tacked on, such as "20,40,56,60". Length Difference counts the number of population sizes off the returned structure is.
- **Mean Absolute Percentage Error** - Measures modeling quality, especially of the next approximated population size. Defined as the percent of the approximation that the absolute value of the difference between the approximation and the error is.
- **Mean Squared Error** - The averaged sum of the squares of differences between approximations in the ground truth and the output of the model.

3.2 VARIANCE OF Z(K)

The next population state in a sequence representing a Birth and Death Process is often trivial. Predicting the Kth population state in a sequence, however, may be more complex. Variance measures the spread of numbers; we avoid small differences in approximation for the next sequence and capture a more complex function with a higher K. We can take the Chapman-Kolmogorov Theorem and show that the number of population sizes, given a continuous-time Markov chain, influences the variance Kolmogoroff (1931); Ross (1995).

$$P_{ij}^{n+m} = \Sigma_{k=0}^{\infty} P_{ik}^{n} P_{kj}^{m}$$
$$P_{zj}^{K} = \Sigma_{k=0}^{\infty} P_{zk}^{n} P_{kj}^{K-n}, n \in (0, K]$$
$$E[Z(K)] = \Sigma_j j P_{zj}^{K}$$
$$E[Z(K)^2] = \Sigma_j j^2 P_{zj}^{K}$$
$$Var(Z(K)) = \Sigma_j j^2 P_{zj}^{K} - (\Sigma_j j P_{zj}^{K})^2$$

Particular Birth and Death Processes may have more precise manipulations of variance. For example, in a Poisson Process, we show that the variance increases with t, and, inherently, K. The mean of a Poisson Distribution is equal to its mean, the average rate, λ. $Var(Z_k) = \lambda t = \lambda K$.

3.3 K VERSUS COMPLEXITY

It is observed that, in some situations, the Kth population size approximation is more precise with more model complexity, indicating that, in some cases, approximating population size with $K > 1$ is a more complicated function than when $K = 1$. We measure the mean squared error of the Kth population size as generated by a Verhulst model.

We observe that the variance of $Z(K)$ also tends to increase as K increases, as shown in Figure 2. A simple linear regression model can predict the next population size for $K = 1$ with just 9.9455 MSE, but this increases to 51.9744, 40.2292, and up to 127.0160 MSE for $K = 2$, $K = 3$, and $K = 9$, respectively.

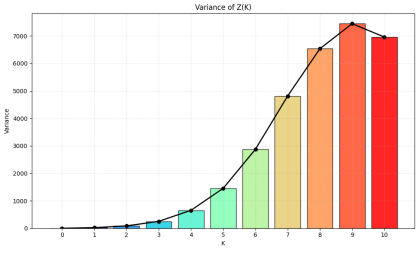


Figure 2: Variance of $Z(K)$ over the initial window from 1000 Verhulst population simulations. The x-axis represents the carrying capacity K. The variance $Var(Z(K))$ tends to increase with K.

This trend persists with a more flexible solution, a decision tree, limiting the flexibility with 6 max leaf nodes. Furthermore, we can reduce the MSE for $K = 1$ to 6.28, and as high as 19.5474, 31.896, and 9.0349 for $K = 2$, $K = 3$, and $K = 9$ with a decision tree with 20 max leaf nodes, indicating that more complexity helps lower MSE. We conclude that, in the context of a Verhulst Model represented by a Birth and Death Process, a higher K adds more complexity to the data that needs a more complex solution.

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$
Linear Regression MSE	9.9455	51.9744	40.2292	52.6022	56.5585	178.4722	179.2485	106.8314	127.0160
DT MSE ($\max_leaf_nodes = 6$)	90.8293	151.0802	317.2582	526.0072	570.6821	599.1498	577.7766	551.1189	456.1711
DT MSE ($\max_leaf_nodes = 20$)	6.2800	10.9817	19.5474	31.1896	38.7243	33.0408	32.7897	25.8401	9.0349

Table 1: Comparison of MSE values with Decision Tree Regressor using different maximum leaf node settings - Verhulst. We observe that a higher K value has less error with more flexibility; this indicates that these problems require more complexity in the model. The Poisson Model can be represented with a low flexibility linear regression model - as error is low, we do not observe the same phenomenon. The Ricker Model, similarly, is better estimated with a more flexible solution such as a decision tree, but does not have the same increase in complexity with K.

3.4 Countering Underfitting with K

Early observations showed much underfitting occurring when applying the self improvement framework to Birth and Death Processes. Large language models during early training were observed to have structure issues with too many separator tokens in the output and learn little about the Birth and Death Process; structures that repeated comma tokens without population sizes approximated, gained a high probability of matching the next token, and were common. Repeating the input structure with no changes was a common fit as well. Extending the length on data from a Verhulst Birth and Death Process on example Meta Llama 3 8B runs reached a high accuracy of 98.55%, but with an average length difference of 1.9277. Nevertheless, on perfect structured examples, the model had a 95 percent confidence interval of $[0.091, 0.107]$ for approximation of its next population size.

When applying the self improvement framework to Birth and Death Processes, we increase the difficulty of the sample x_i each round by increasing its length. We denote K as the number of population sizes added to x_i each round. We see that we may increment length by K such that K is a parameter that influences the variance of population sizes being estimated. Simulations and intuition show that variance increase with K; there are less trivial results, intuitively, with $K > 1$. The data is therefore more complex and takes advantage of the high flexibility of a large language model. Through the law of bias variance tradeoff James et al. (2013), we counter underfitting by increasing K, as a very flexible model such as a large language model can take advantage of more complexity in the data.

3.5 Data

We generate 1,250,000 Birth and Death Process simulations and their next state through BirDePy.

3.6 Procedure

We use our datasets of 1,250,000 simulations, and select 3,000 random simulations to create $\mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4 ... \mathcal{D}_{35}$, extending the lengths for 35 self improvement rounds.

4 Results

4.1 Benchmarks

We benchmark Birth and Death Processes on existing T5 architectures trained on NVIDIA A40 GPU's Raffel et al. (2020). We show that length generalization remains a problem on these tasks Anil et al. (2022). The highest performance benchmark on the Verhulst population data reaches 84.33% when tested against in distribution tasks. Out of distribution data of the highest length test

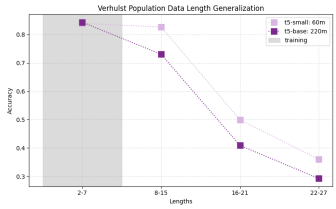


Figure 3: Verhulst

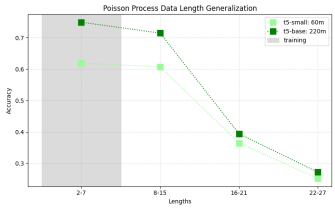


Figure 4: Poisson

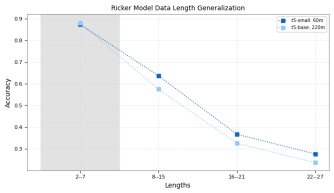


Figure 5: Ricker

dataset lowers accuracy to 35% and 29%. Length difference, measuring structure, starts close to 0 for in distribution data and grows quickly.

Model	Length	Accuracy	Length Diff	Avg MSE
T5-small	2–7	0.8393	0.02	19824
T5-small	8–15	0.8262	1.96	85871
T5-small	16–21	0.4991	8.76	254912
T5-small	22–27	0.3599	14.79	339462
T5-base	2–7	0.8433	0.01	8484
T5-base	8–15	0.7302	2.39	106443
T5-base	16–21	0.4087	9.60	294032
T5-base	22–27	0.2924	15.71	367654

Verhulst

Model	Length	Accuracy	Length Diff	Avg MSE
T5-small	2–7	0.6187	0.09	68843
T5-small	8–15	0.6070	2.42	56266
T5-small	16–21	0.3648	9.28	67963
T5-small	22–27	0.2520	15.43	70400
T5-base	2–7	0.7489	0.45	57237
T5-base	8–15	0.7143	2.64	64646
T5-base	16–21	0.3943	9.83	64678
T5-base	22–27	0.2720	16.15	70023

Poisson

Model	Length	Accuracy	Length Diff	Avg MSE
T5-small	2–7	0.8728	0.08	54.2901
T5-small	8–15	0.6365	5.713	685.2705
T5-small	16–21	0.3665	18.9	2097.8192
T5-small	22–27	0.2760	30.265	28680.1231
T5-base	2–7	0.8793	0.034	28.3708
T5-base	8–15	0.5743	6.91	3096.0870
T5-base	16–21	0.3255	21.112	2147.4838
T5-base	22–27	0.23603	33.503	2560.6815

Ricker

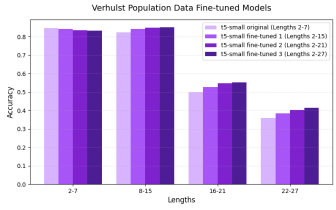
Table 2: Performance of T5-small and T5-base models across length brackets for the Verhulst, Poisson, and Ricker datasets.

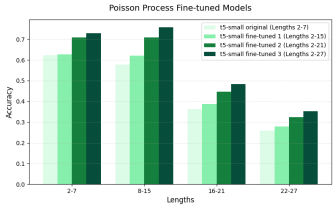
4.2 Prompting

Prompting a large language model with details can yield human level performance on offshoot tasks such as estimating the next number in a Birth and Death Process. GPT-4 exhibits human level performance on various academic and professional benchmarks, and GPT-5 and GPT-6 Astra perform even better Achiam et al. (2023). A random sample of 100 from the testing data set on GPT-6 yield an MSE of 19,003.2 and a MAPE of 56.28% when given the prompt "Find the next number", while the same sample yields an MSE of 1,674.46 given the phrase "Find the next number from a Verhulst model" and a MAPE of 21.698%; reasoning ability comes from public internet text instead of BirDePy and answers require extraction from natural text form. Both prompts respect carrying capacity in our GPT-6 trials. Prompting delivers task specific details to answer longer problems that may not scale well, but nevertheless delivers human performance.

4.3 Fine Tuning

Fine tuning has been used a method of length generalization on other tasks Anil et al. (2022). It significantly improves performance on appropriate lengths when approximating Birth and Death Processes. Performance improves on a length bracket by training for a single epoch on as low as 1000 observations drawn from its distribution. Fine tuning on longer datasets increases accuracy from 35.99% to 41.32% on the longest length bracket for the Verhulst Population dataset. On the original in distribution dataset, the original benchmark has an accuracy of 84.54%; the same model then fine tuned on lengths 8-15 has an accuracy of 84.15%, an accuracy of 83.60% when further fine tuned on lengths 16-21, and an accuracy of 83.25% when further fine tuned on lengths 22-27. We see similar results when repeating the process on data from a Poisson and Ricker Birth and Death Process.





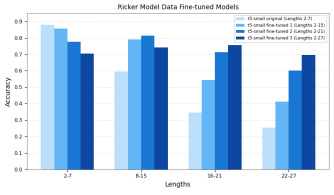


Figure 6: The impact of fine-tuning a large language model on the Verhulst population model, Poisson process, and Ricker model. We see that we can generally increase accuracy on the longest length brackets with fine tuning.

4.4 Underfitting Experiment on the Self Improvement Framework

It is observed that raising K helps defeat manually observed underfitting examples in example situations. We cause underfitting by limiting the data to 12,500 simulations, and use the self improvement framework to extend up to length 40. Accuracy and the proportion with perfect length are metrics that show that our model learned the structure needed to self improve; observed underfitting is best measured by rows with incorrect structure before modeling quality is even considered. Using confidence intervals defined using the standard normal approximation $\hat{p} \pm 1.96\sqrt{\frac{\hat{p}(1-\hat{p}}{n}}$, we achieve a statistically significant increase in higher accuracy for Ricker, Verhulst, and Poisson Models with a higher K value. We achieve a statistically significant increase in the percentage of rows with perfect structure with a higher K value for all Birth and Death Processes. We therefore conclude that the added complexity with a higher K value counters underfitting.

Table 3: Accuracy for different population processes and values of K, with 95% confidence intervals.

Ricker ($K = 1$)	Ricker ($K = 3$)	Verhulst ($K = 1$)	Verhulst ($K = 3$)	Poisson ($K = 1$)	Poisson ($K = 3$)
83.1537%	85.2839%	83.5977%	84.8193%	90.3822%	92.2464%
[82.9240%, 83.3834%]	[84.9016%, 85.6662%]	[83.3704%, 83.8250%]	[84.4321%, 85.2065%]	[90.1401%, 90.6243%]	[91.8557%, 92.6371%]

Table 4: Perfect Structure Proportion for different population processes and values of K, with 95% confidence intervals.

Ricker ($K = 1$)	Ricker ($K = 3$)	Verhulst ($K = 1$)	Verhulst ($K = 3$)	Poisson ($K = 1$)	Poisson ($K = 3$)
6.44%	10.35%	6.22%	10.67%	12.91%	31.10%
[6.2893%, 6.5907%]	[10.0214%, 10.6786%]	[6.0718%, 6.3682%]	[10.3369%, 11.0031%]	[12.6219%, 13.1981%]	[30.4214%, 31.7786%]

4.5 Self Improvement Framework

We use the self improvement framework to extend the length of large language models approximating birth and death processes.

4.5.1 Verhulst

We extend the length of simulations of a Verhulst Model approximated by language models. The technique is overwhelmingly successful, producing correct results up to a length of 40. On the T5-small, we see a statistically significant decrease in mean average percent error of our approximations in comparison to our BirDepy approximations by increasing K. The drop in mean average percent error persists when comparing confidence intervals by r as well as length bracket. Almost all approximations achieve over 99% model outputs with perfect structure.

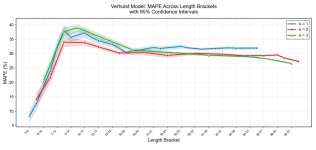


Figure 7: T5-small

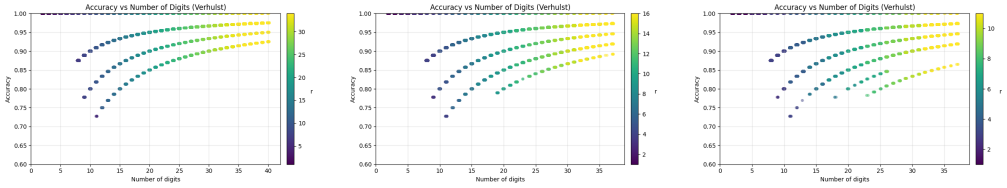


Figure 8: Results on the Verhulst Process data. As done in the original self improvement framework paper from 2025, we enable a model trained on 2 to 7 digits to 40 digits and beyond. We show an ablation study by varying K as 1 (Left), 2 (Middle), and 3 (Right) on the T5-small.

The T5-base also shows improvement with a higher value of K, albeit some tradeoffs. Modeling quality at high length brackets is better at

Model	k	Label/R 2	3	4	5	6	7	8	9	10	11
						Verhulst					
Percent Perfect Structure	1	100.00	100.00	100.00	100.00	99.87	99.47	99.20	98.80	98.73	98.47
MAPE (%)		8.30	12.62	18.40	23.79	32.08	38.00	35.62	37.14	35.64	34.45
SD (%)		49.56	58.62	49.62	54.01	59.67	54.27	41.29	42.57	34.65	28.17
95% CI (%)		[6.53,10.07]	[10.52,14.72]	[16.62,20.18]	[21.86,25.72]	[29.94,34.22]	[36.06,39.94]	[34.14,37.10]	[35.62,38.66]	[34.40,36.88]	[33.44,35.46]
Length Bracket		3–8	4–9	5–10	6–11	7–12	8–13	9–14	10–15	11–16	12–17
Percent Perfect Structure	2	100.00	100.00	99.47	98.93	98.60	99.67	99.83	99.40	99.23	98.80
MAPE (%)		14.10	21.62	33.93	33.78	32.31	30.18	30.28	29.33	30.04	29.77
SD (%)		65.59	48.92	44.58	25.08	17.34	15.01	23.90	18.86	13.85	8.69
95% CI (%)		[11.75,16.45]	[19.87,23.37]	[32.33,35.53]	[32.88,34.68]	[31.69,32.93]	[29.64,30.72]	[29.42,31.14]	[28.65,30.01]	[29.54,30.54]	[29.46,30.08]
Length Bracket		4–9	6–11	8–13	10–15	12–17	14–19	16–21	18–23	20–25	22–27
Percent Perfect Structure	3	100.00	99.33	98.10	99.77	100.00	99.67	99.43	99.60	99.90	100.00
MAPE (%)		20.23	37.59	38.89	35.04	31.22	30.25	29.39	28.89	27.97	26.53
SD (%)		54.27	44.05	34.49	27.68	17.35	10.00	7.27	6.60	6.27	5.31
95% CI (%)		[18.29,22.17]	[36.01,39.17]	[37.66,40.12]	[34.05,36.03]	[30.60,31.84]	[29.89,30.61]	[29.13,29.65]	[28.65,29.13]	[27.75,28.19]	[26.34,26.72]
Length Bracket		5–10	8–13	9–16	12–19	15–22	18–25	21–28	24–31	27–34	30–37

Table 5: Comparison of Verhulst Model approximations on the T5-small. Results yield an accuracy of 99.2881% and an average length difference of 0.0138 for a K of 1, 98.24% and 0.0176 for a K of 2, and 97.68% 0.0232 for a K of 3.

$K = 1$ for our results with perfect structure, with a 95% confidence interval of $[34.35, 35.57]$ for the last length bracket, compared to $[37.72, 39.90]$ for $K = 3$. We see a statistically significant drop in perfect structure for higher length brackets; the higher length brackets for a K value of 1 only have 84% perfect correct, while K values of 2 and 3 stay above 96%. We show full results in the Appendix.

4.6 Poisson and Ricker

We repeat the procedure on the Poisson Process and Ricker Model. We see overall success in approximating; the Poisson Process is a simple pattern and we do not observe a need to adjust K to meet high model complexity; the Ricker process is fit well with a K of 1. We have an accuracy of 98.8833% and length difference of 0.0150 for a K of 1, 98.8758% and 0.0081 for 2, and 98.791% and 0.0140 for 3 with the Poisson Process. For the Ricker Model, we have an accuracy of 99.0917% and a length difference of 0.1356 for a K of 1, 98.7273% and 0.2193 for a K of 2, and 98.6562% and 0.1619 for a K of 3. MAPE is adequate, staying below 20% for most rounds. See the Appendix for full results.

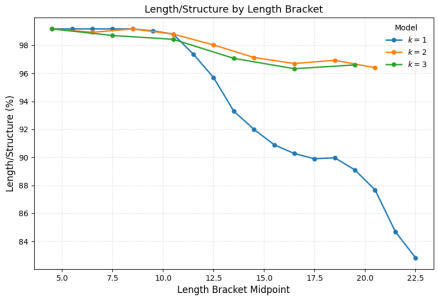


Figure 9: Percent of Verhulst Model Outputs with Perfect Structure By K on T5-Base.

4.7 Parameter Estimation of Self-Improvement Models Through State Models

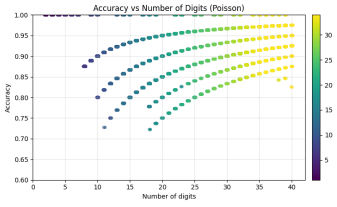


Figure 10: Results on the Poisson Process data.

The quality of large language models approximating a markov chain can be measured through mean squared error and similar metrics against a test dataset that provides a ground truth. Relative differences against the ground truth results provides an estimate of how well the large language model performs. However, the results of a Birth and Death Process may vary with different transition probabilities, and predictions that are completely different from the ground truth may be more accurate than one not far off in mean squared error. Estimating the parameters involved in a Birth and Death Process can prove to be a more powerful manner of measuring the generalization capabilities of M_r. A state model is perfect to this end Shumway & Stoffer (2006); King et al. (2016). A continuous-time Markov Chain that dictates the population in a Birth and Death Process can be implemented to model the internals of a system inside a state model. Inside of a state model, this Hidden Markov Chain will dictate the latent state, X_n, and the population can be derived from X_n as the observed value y_n.

After approximating 10 Verhulst population sizes, a Profile Plot shows that the maximum likelihood growth parameter rate is consistent with the parameter rate used to generate data, showing that $\theta = \{\beta\} = \{0.75\}$ has a high likelihood when used with the state model, and is not far off from the maximum likelihood estimator from the state-space model $\hat{\theta} = \{\hat{\beta}\} = \{0.84\}$ Bhadra et al. (2011). We repeat with Poisson; $\theta = \{\lambda\} = \{3\}$ has a high likelihood when used with the state model, and is not far off from the maximum likelihood estimator $\hat{\theta} = \{\hat{\lambda}\} = \{2.1\}$.

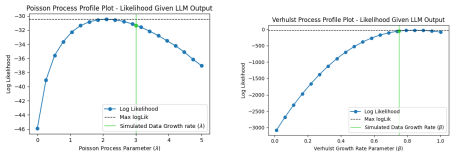


Figure 11: Verhulst and Poisson Process Profile Plots. The maximum likelihood parameter estimates are 2.1 and 0.84, respectively.

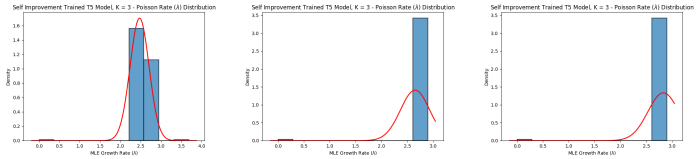


Figure 12: 2-7, 9-11, and 16-21 length λ distributions, from left to right. $\hat{\lambda} = 2.3684$ (Left), $\hat{\lambda} = 2.6316$ (Middle), $\hat{\lambda} = 2.8947$ (Right). We take outputs from the self-improvement trained large language model for the Poisson process, with data generated from $\lambda = 3$, using appropriate length datasets, and obtain the maximum likelihood parameter estimates to create a distribution.

4.8 Application to Bird Populations

We train a T5-small model on Verhulst Population data with $\beta = 0.35, \nu = 0.3$. Figure 13 shows an estimate of the Aransas-Wood Buffalo Whooping Crane abundance from 1942 through 2010 Strobel et al. (2012); the training distribution is only 2-7 years.

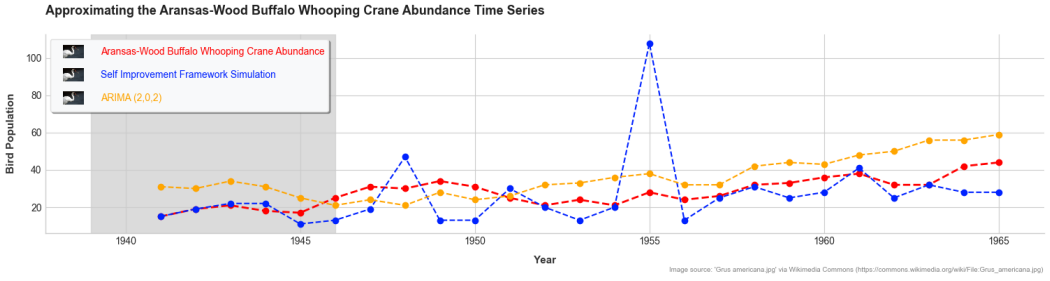


Figure 13: Aransas-Wood Buffalo Whooping Crane abundance simulated beyond an initial training distribution (grey). The simulation numbers are recorded through the most recent M_r; we use teacher forcing on the ARIMA and Self Improvement Framework Simulation numbers to obtain MSE of 140.29 and 115.47 when removing the outlier at 1955, respectively.

5 Conclusion

Birth and Death Processes are modeled well by large language models. With the self improvement framework, length generalization becomes a solved problem for some data. Self-improvement trained models show that large language models can extrapolate length handling and output structure from in-distribution instances from all tested Birth and Death Processes, and successfully apply them to higher lengths.

ACKNOWLEDGMENTS

LLM's have been used in this project to conduct help collecting papers during literature review, locate available data online, debug and assist during coding, primarily of evaluation scripts, and assist in latex and visualization formatting. All code is primarily human written, with shortcuts (MSE calculating methods, etc.) provided as assisted by language models.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 38546–38556, 2022.

Nicolas Bacaër. Verhulst and the logistic equation (1838). In *A Short History of Mathematical Population Dynamics*, pp. 35–39. Springer London, London, 2011.

Anindya Bhadra, Edward L Ionides, Karina Laneri, Mercedes Pascual, Menno Bouma, and Ramesh C Dhiman. Malaria in northwest india: Data analysis via partially observed stochastic differential equation models driven by lévy noise. *Journal of the American Statistical Association*, 106(494):440–451, 2011.

Forrest W Crawford and Marc A Suchard. Transition probabilities for general birth–death processes with applications in ecology, genetics, and evolution. *Journal of mathematical biology*, 65(3): 553–580, 2012.

Charles R Doss, Marc A Suchard, Ian Holmes, Midori Kato-Maeda, and Vladimir N Minin. Fitting birth-death processes to panel data with applications to bacterial dna fingerprinting. *The Annals of Applied Statistics*, 7(4):2315, 2013.

Sophie Hautphenne and Brendan Patch. Birth-and-death processes in Python: The BirDePy package. *Journal of Statistical Software*, 111:1–54, 2024.

Lam Si Tung Ho, Jason Xu, Forrest W Crawford, Vladimir N Minin, and Marc A Suchard. Birth/birth-death processes and their computable transition probabilities with biological applications. *Journal of mathematical biology*, 76(4):911–944, 2018.

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, et al. *An introduction to statistical learning: with applications in R*, volume 103. Springer, 2013.

Aaron A King, Dao Nguyen, and Edward L Ionides. Statistical inference for partially observed markov processes via the r package pomp. *Journal of Statistical Software*, 69:1–43, 2016.

Andrei Kolmogoroff. Über die analytischen methoden in der wahrscheinlichkeitsrechnung. *Mathematische Annalen*, 104(1):415–458, 1931.

Nayoung Lee, Ziyang Cai, Avi Schwarzchild, Kangwook Lee, and Dimitris Papailiopoulos. Self-improving transformers overcome easy-to-hard and length generalization challenges. *arXiv preprint arXiv:2502.01612*, 2025.

Artem S Novozhilov, Georgy P Karev, and Eugene V Koonin. Biological applications of the theory of birth-and-death processes. *Briefings in bioinformatics*, 7(1):70–85, 2006.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

William Edwin Ricker. Linear regressions in fishery research. *Journal of the fisheries board of Canada*, 30(3):409–434, 1973.

Sheldon M. Ross. *Stochastic Processes*. John Wiley & Sons, 1995.

Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Its Applications: With R Examples*. Springer New York, New York, NY, 2006.

B Strobel, MJ Butler, and G Harris. Aransas-wood buffalo whooping crane abundance survey (2011-2012). *Aransas National Wildlife Refuge, US Fish and Wildlife Service, Austwell, Texas, USA*, 2012.

William J Thompson. Poisson distributions. *Computing in Science & Engineering*, 3(3):78–82, 2001.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. In *International Conference on Learning Representations*, volume 2024, pp. 15898–15926, 2024.

A APPENDIX

A.1 METRICS BY SELF IMPROVEMENT ROUND

Table 6: Comparison of T5-small with Poisson Models Across Labels

		Label/R									
Model	K/r	2	3	4	5	6	7	8	9	10	11
		Poisson									
Percent Perfect Structure	1	99.17	99.17	99.17	99.07	97.30	97.33	92.37	88.57	85.77	95.33
95% CI (Structure)		[98.84,99.49]	[98.84,99.49]	[98.84,99.49]	[98.72,99.41]	[96.72,97.88]	[96.76,97.91]	[91.42,93.32]	[87.43,89.71]	[84.52,87.02]	[94.58,96.09]
MAPE (%)		12.05	12.85	13.09	14.53	13.24	16.30	14.67	10.66	11.43	10.32
SD (%)		16.39	70.77	81.44	99.49	83.20	112.90	83.07	22.07	27.69	34.02
95% CI (MAPE)		[11.46,12.64]	[10.31,15.38]	[10.18,16.01]	[10.97,18.10]	[10.26,16.21]	[12.26,20.35]	[11.70,17.65]	[9.87,11.45]	[10.43,12.42]	[9.10,11.53]
Length Bracket		3–8	4–9	5–10	6–11	7–12	8–13	9–14	10–15	11–16	12–17
Percent Perfect Structure	2	99.17	98.90	97.57	92.10	97.50	98.37	99.17	99.17	98.87	98.80
95% CI (Structure)		[98.84,99.49]	[98.53,99.27]	[97.02,98.12]	[91.13,93.07]	[96.94,98.06]	[97.91,98.82]	[98.84,99.49]	[98.84,99.49]	[98.49,99.25]	[98.41,99.19]
MAPE (%)		12.90	14.52	17.13	20.75	16.13	14.47	19.12	20.66	19.26	20.30
SD (%)		70.78	99.50	114.63	131.67	93.85	63.87	84.48	78.51	44.10	37.82
95% CI (MAPE)		[10.37,15.43]	[10.96,18.08]	[13.03,21.24]	[16.04,25.46]	[12.77,19.49]	[12.19,16.76]	[16.10,22.15]	[17.85,23.47]	[17.68,20.84]	[18.95,21.66]
Length Bracket		4–9	6–11	8–13	10–15	12–17	14–19	16–21	18–23	20–25	22–27
Percent Perfect Structure	3	99.17	99.17	97.37	89.97	98.40	98.77	98.57	98.67	98.93	98.50
95% CI (Structure)		[98.84, 99.49]	[98.84, 99.49]	[96.79, 97.94]	[88.89, 91.04]	[97.95, 98.85]	[98.37, 99.16]	[98.14, 98.99]	[98.26, 99.08]	[98.57, 99.30]	[98.07, 98.93]
MAPE (%)		15.60	13.01	17.26	19.29	16.68	21.61	26.84	32.79	40.63	42.52
SD (%)		26.97	81.43	114.80	122.13	65.38	86.84	102.50	112.25	138.45	134.27
95% CI (MAPE)		[14.63,16.56]	[10.09,15.93]	[13.15,21.37]	[14.92,23.67]	[14.34,19.02]	[18.50,24.72]	[23.17,30.50]	[28.77,36.81]	[35.67,45.58]	[37.71,47.33]
Length Bracket		5–10	8–13	11–16	14–19	17–22	20–25	23–28	26–31	29–34	32–37

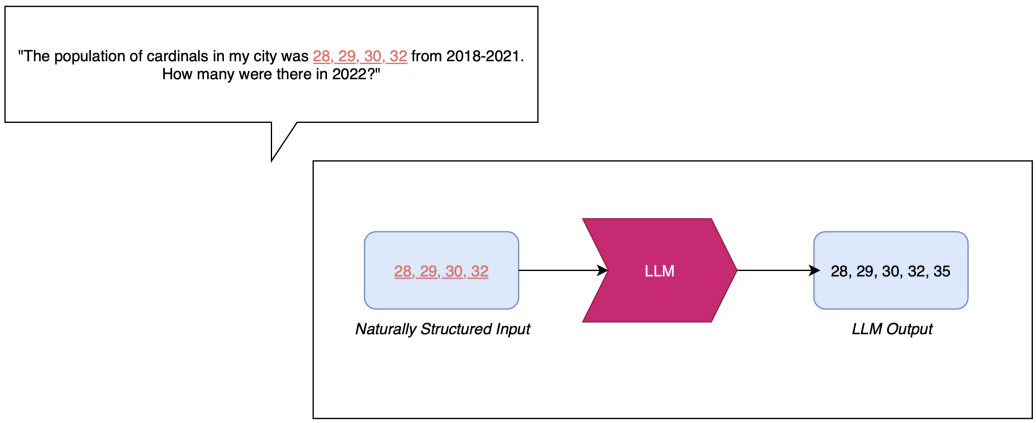
Table 7: Comparison of T5-small with Ricker Models Across Labels.

		Label/R									
Model	K/r	2	3	4	5	6	7	8	9	10	11
		Ricker									
Percent Perfect Structure	1	99.17	99.17	99.17	99.00	98.67	98.27	98.10	97.43	96.47	96.30
95% CI (Structure)		[98.84,99.49]	[98.84,99.49]	[98.84,99.49]	[98.64,99.36]	[98.26,99.08]	[97.80,98.73]	[97.61,98.59]	[96.87,98.00]	[95.81,97.13]	[95.62,96.98]
MAPE (%)		12.17	11.98	11.70	10.00	10.84	12.98	14.98	15.26	16.29	18.27
SD (%)		31.42	30.79	24.53	16.41	18.79	16.84	25.73	11.51	11.46	11.04
95% CI (MAPE)		[11.04,13.30]	[10.88,13.09]	[10.82,12.58]	[9.41,10.59]	[10.16,11.51]	[12.37,13.58]	[14.06,15.91]	[14.85,15.68]	[15.87,16.70]	[17.87,18.66]
Length Bracket		3–8	4–9	5–10	6–11	7–12	8–13	9–14	10–15	11–16	12–17
Percent Perfect Structure	2	99.17	98.90	98.37	96.50	96.20	92.30	89.43	84.13	81.00	76.47
95% CI (Structure)		[98.84,99.49]	[98.53,99.27]	[97.91,98.82]	[95.84,97.16]	[95.52,96.88]	[91.35,93.25]	[88.33,90.53]	[82.83,85.44]	[79.60,82.40]	[74.95,77.98]
MAPE (%)		11.22	10.02	12.65	15.24	18.68	21.67	28.32	37.98	42.71	43.87
SD (%)		25.87	16.42	11.63	11.23	10.85	10.88	14.77	16.22	15.54	14.92
95% CI (MAPE)		[10.29,12.15]	[9.43,10.61]	[12.24,13.07]	[14.84,15.64]	[18.29,19.07]	[21.28,22.06]	[27.79,28.85]	[37.40,38.56]	[42.15,43.26]	[43.33,44.40]
Length Bracket		4–9	6–11	8–13	10–15	12–17	14–19	16–21	18–23	20–25	22–27
Percent Perfect Structure	3	98.50	97.77	95.83	93.50	89.33	87.07	80.00	69.43	62.23	47.23
95% CI (Structure)		[98.07,98.93]	[97.24,98.30]	[95.12,96.55]	[92.62,94.38]	[88.23,90.44]	[85.87,88.27]	[78.57,81.43]	[67.78,71.08]	[60.50,63.97]	[45.45,49.02]
MAPE (%)		10.48	12.55	16.37	21.53	33.51	41.14	39.87	33.83	22.77	20.62
SD (%)		13.69	11.57	11.47	11.30	16.67	15.28	13.77	16.79	12.65	7.89
95% CI (MAPE)		[9.99,10.97]	[12.13,12.97]	[15.96,16.78]	[21.13,21.94]	[32.92,34.11]	[40.59,41.69]	[39.37,40.36]	[33.22,34.43]	[22.32,23.23]	[20.33,20.90]
Length Bracket		5–10	8–13	11–16	14–19	17–22	20–25	23–28	26–31	29–34	32–37

Table 8: Comparison of T5-Base with Verhulst Models Across Labels

	Label/R					
Model	K/r	2	3	4	5	6
T5-base + Verhulst						
Percent Perfect Structure	1	99.17	99.17	99.17	99.17	99.03
95% CI (Structure)		[98.84,99.49]	[98.84,99.49]	[98.84,99.49]	[98.84,99.49]	[98.68,99.38]
MAPE (%)		9.63	14.07	17.93	22.58	27.85
SD (%)		36.70	40.16	38.30	40.39	39.82
95% CI (MAPE)		[8.31,10.95]	[12.63,15.51]	[16.55,19.30]	[21.13,24.03]	[26.41,29.28]
Length Bracket		3–8	4–9	5–10	6–11	7–12
Percent Perfect Structure	2	98.93	99.17	98.80	98.03	97.13
95% CI (Structure)		[98.57,99.30]	[98.84,99.49]	[98.41,99.19]	[97.54,98.53]	[96.54,97.73]
MAPE (%)		15.77	25.77	40.86	43.21	43.71
SD (%)		49.25	50.59	57.53	51.02	44.60
95% CI (MAPE)		[14.00,17.54]	[23.95,27.59]	[38.80,42.93]	[41.37,45.04]	[42.11,45.31]
Length Bracket		4–9	6–11	8–13	10–15	12–17
Percent Perfect Structure	3	98.70	98.43	97.07	96.33	96.60
95% CI (Structure)		[98.29,99.11]	[97.99,98.88]	[96.46,97.67]	[95.66,97.01]	[95.95,97.25]
MAPE (%)		19.80	40.05	44.25	42.51	38.81
SD (%)		48.77	55.18	41.06	36.67	30.34
95% CI (MAPE)		[18.05,21.55]	[38.06,42.03]	[42.77,45.73]	[41.19,43.83]	[37.72,39.90]
Length Bracket		5–10	8–13	11–16	14–19	17–22

A.2 Example Usage - Conversational Agent



A.3 Complexity Vs K

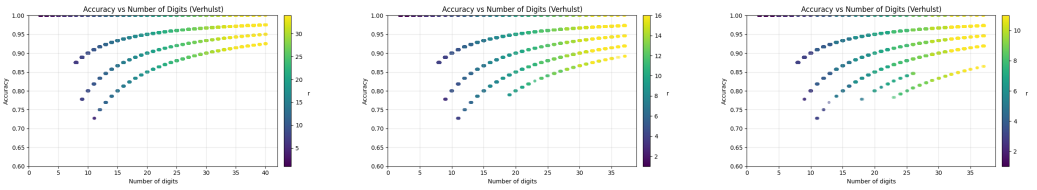
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$
Linear Regression MSE	16.5413	12.0353	15.6110	5.8677	5.0821	7.3671	7.5296	8.2707	6.2597
DT MSE ($\text{max_leaf_nodes} = None$)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
DT MSE ($\text{max_leaf_nodes} = 6$)	24.2423	15.4375	11.7043	7.9668	8.7068	4.2310	4.1264	4.0985	2.8854
DT MSE ($\text{max_leaf_nodes} = 20$)	1.5949	0.9959	0.6611	0.5773	0.4684	0.3546	0.6174	0.4498	0.5033

Table 9: Comparison of MSE values with Decision Tree Regressor using different maximum leaf node settings - Ricker.

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$
Linear Regression MSE	0	0	0	0	0	0	0	0	0
DT MSE ($\text{max_leaf_nodes} = None$)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
DT MSE ($\text{max_leaf_nodes} = 6$)	30.3032	33.6951	33.0870	31.7698	31.0199	31.3673	33.1389	33.1299	31.7537
DT MSE ($\text{max_leaf_nodes} = 20$)	2.4536	2.5738	2.4022	2.5095	2.3754	2.4009	2.5358	2.5095	2.4315

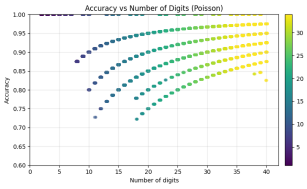
Table 10: Comparison of MSE values with Decision Tree Regressor using different maximum leaf node settings - Poisson.

A.4 Self Improvement Framework Diagrams

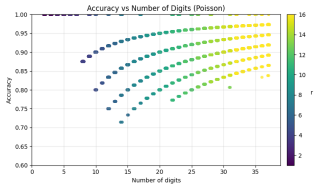


(a) K = 1. (b) K = 2. (c) K = 3.

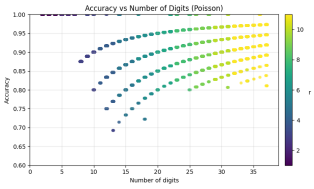
Figure 14: Verhulst, T5-small



(a) K = 1.

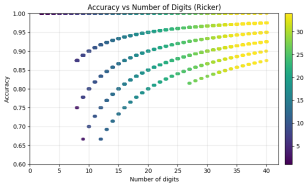


(b) K = 2.

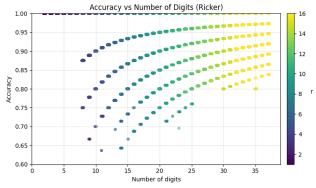


(c) K = 3.

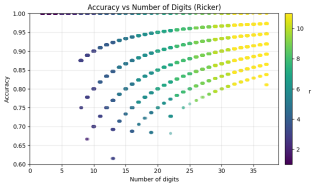
Figure 15: Poisson, T5-small



(a) K = 1.

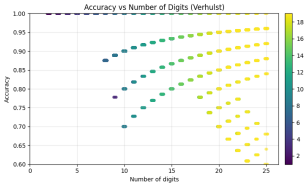


(b) K = 2.

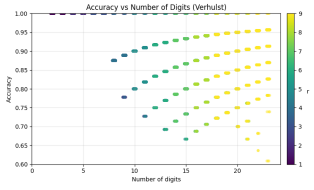


(c) K = 3.

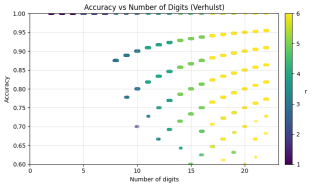
Figure 16: Ricker, T5-small



(a) K = 1.



(b) K = 2.



(c) K = 3.

Figure 17: Verhulst, T5-base