Approximating Longer Birth and Death Processes with Large Language Models and the Self Improvement Framework

Anonymous ACL submission

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 generalizations to the problem, including extending the self improvement framework to data from birth and death processes to benchmark performance. To overcome 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.

1 Introduction

A Birth and Death Process is formally defined as a continuous-time Markov chain with states $\{i \in Z \mid 0 \leq i \leq n\}$ 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 and 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.

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.

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? 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.

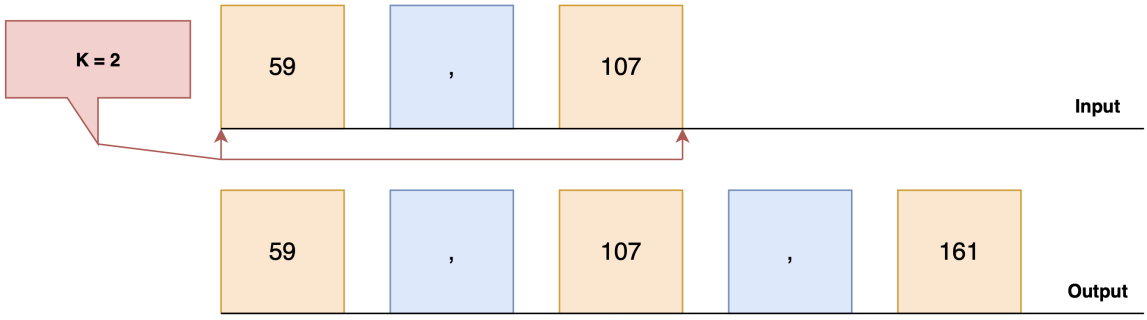


Figure 1: A Birth and Death Process, as input into a self-improvement trained language model. The input consists of a sequence containing two population sizes and separator tokens in between; part of the task is for the model to understand the structure and to put separator tokens in between. In this 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.

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.

3 Background Knowledge

3.1 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.

3.2 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 and 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.

3.3 Verhulst Model

The Verhulst model is also known as the logistic growth model. It was introduced by Pierre Francois Verhulst in 1838 (Bacaïer, 2011). It simulates population growth that approaches a carrying capacity. We define the growth per unit of time.

$$\frac{dP}{dt} = rP(1 - \frac{P}{K})$$

3.4 Poisson

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.

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

4 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.

Underfitting is a problem observed during training; large language models, even small ones, contain hundreds of millions of parameters. Population transitions are difficult to learn, and representing the structure of a Markov Process $Z(t)$ is a large component of the problem; separation tokens in between population sizes need to be learned by the large language model. 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. We can add variance to the population sizes predicted so that the problem is more complex. For example, instead of including trivial observations that lead to the model learning how to to game the training objective with little beneficial learning, we increase the variance so that the final population size is more difficult to learn. More structure to learn and a more difficult population size problem prevent the large language model from reaching a trivial solution that over-optimizes to the structure, for example. 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).

$$P_{ij}^{n+m} = \Sigma_{k=0}^{\infty} P_{ik}^{n} P_{kj}^{m}$$

$$P_{zj}^{K} = \Sigma_{k=0}^{\infty} P_{zk}^{n} P_{kj}^{K-n}, n \in (0, K]$$

$$E[Z(K)] = \Sigma_j j P_{zj}^{K}$$

$$E[Z(K)^2] = \Sigma_j j^2 P_{zj}^{K}$$

$$Var(Z(K)) = \Sigma_j j^2 P_{zj}^{K} - (\Sigma_j j P_{zj}^{K})^2$$

In the Self Improvement Framework, we increase the difficulty of the sample x_i by increasing its length. We denote K as the number of population sizes added to x_i each self improvement 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. Through the law of bias variance tradeoff (James et al., 2013), we counter underfitting by increasing K.

Particular Birth and Death Processes may have more precise manipulations of variance. For example, in a Poisson Process, we show that the variance

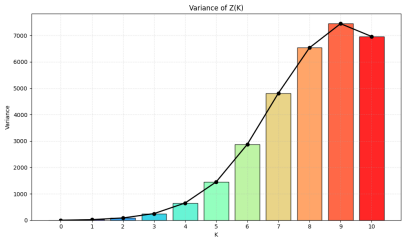


Figure 2: Variance of $Z(K)$ from the initial window from 1000 Verhulst population data simulations; K is the x-axis. $Var(Z(K))$ increases with most K.

increases with t, and, inherently, K. The mean of a Poisson Distribution is equal to its mean, the average rate, λ. $Var(Z_k) = \lambda t = \lambda K$.

4.1 Data

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

5 Results

5.1 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 dataset lowers accuracy to 35% and 29%.

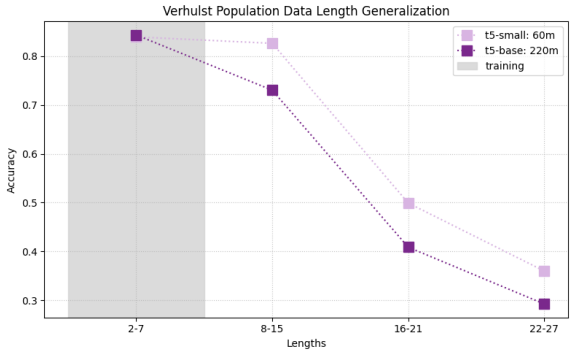


Figure 3: Length graphed against accuracy for two T5 models on the Verhulst population dataset. Accuracy falls with longer inputs.

Length generalization remains a problem for approximating a Poisson Process too. Accuracy starts out as high as 74.89% on the T5-base model on test data drawn from the initial distribution. Accuracy falls to the $20 - 30\%$ range by the lowest length bracket. Table 2 features these metrics.

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

Table 1: Verhulst Population data metrics on benchmark t5 large language models.

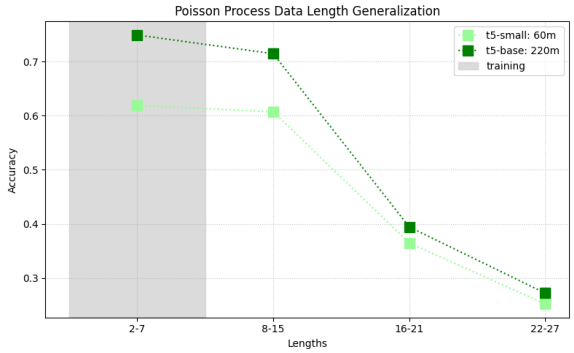


Figure 4: Length graphed against accuracy for two T5 models on the Poisson Process dataset. Accuracy falls with longer inputs.

5.2 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 (Achiam et al., 2023). A random sample of 10 from the testing data set on GPT-4 yield an MSE of 910.7 when given the prompt "Find the next number", while the same sample yields an MSE of 632.8 given the phrase "Find the next number from a Verhulst model"; these come from public internet text instead of BirDePy and require extraction. Given lengthy inputs, the latter phrase also helps GPT-4 respect carrying capacity. The former phrase can yield a 10% difference when over-

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

Table 2: Poisson Process Metrics on benchmark T5 large language models.

shooting attributed to carrying capacity. Prompting delivers task specific details that may not scale well, but nevertheless delivers human performance.

5.3 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.

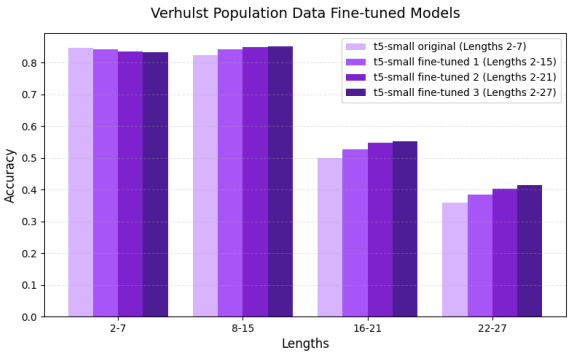


Figure 5: The impact of fine tuning on a large language model on a Verhulst Population Dataset. The original large language model retains the highest accuracy on the initial training distribution. Additional training on larger lengths reduces the accuracy drop in the largest length bracket.

Repeating the process on data from a Poisson Birth and Death Process, on the original in distribution dataset, the original benchmark has an accuracy of 62.21%; the same model then fine tuned on lengths 8-15 has an accuracy of 62.69%, an accuracy of 70.84% when further fine tuned on lengths 16-21, and an accuracy of 73.01% when further fine tuned on lengths 22-27.

5.4 Self Improvement Framework

The Self Improvement Framework enables models to solve problems beyond their initial training distribution; we extend it to the task of predicting the next population size in a Birth and Death Process, using data generated from processes such as the Verhulst Model and a Poisson Process.

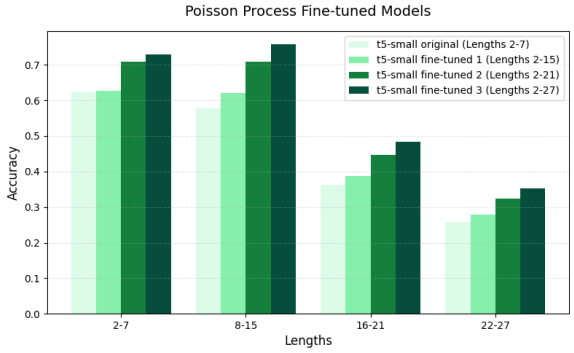


Figure 6: The impact of fine tuning on a large language model for a Poisson Process. The largest length bracket with the most fine tuning has 35.19% accuracy compared to 25.77% without it.

Each round r, we take data from the previous round $r-1$ and extend the length of the input prompt to include the next K population sizes. We then predict the $(K+1)^{th}$ population size from the previous model M_{r-1}, and relabel the dataset.

While the self improvement framework has been applied to tasks such as arithmetic, string copying, and reverse addition, stochastic processes such as a Verhulst model approximated by a Birth and Death Process chain can be more complex. Models with just one epoch and $K=1$ struggle with the structure of the text as well recognizing patterns. After testing instances with lengths of 1 to 3, Figure 6 shows that models drop in average accuracy per token to below 20%. While there is some recovery with more training time reached at longer lengths, the objective is often fit to; ex, returning all separator tokens.

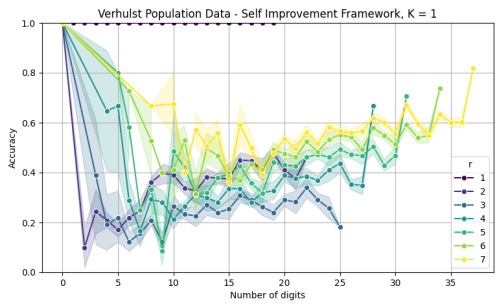


Figure 7: The self improvement framework applied to the Verhulst population data.

By adding more population sizes, we increase the variance of the population sizes being deter-

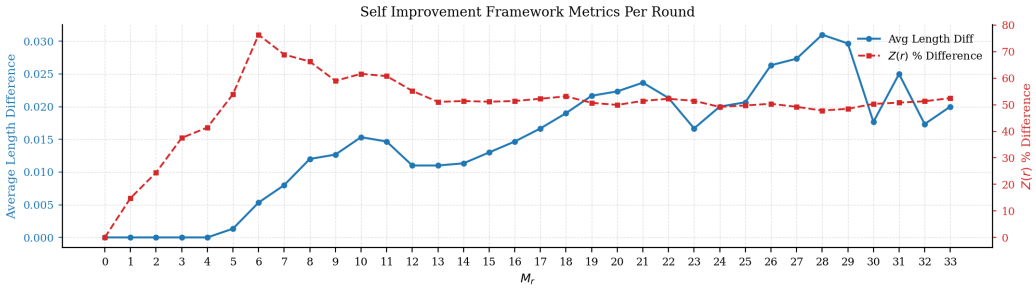


Figure 8: Self Improvement Framework time series metrics per model M_r on Verhulst population data with $K = 1$. Average percent difference from BirDePy ground truth estimates stabilize while length stays in range of perfection. For $K = 3$, runs yield 36.79% to 26.91% average percent difference for the last round.

mined, which helps remove bias seen with lower K values. Training accuracy per token increases to 91.7%. We see many examples where the length and structure are recognized, and the transition matrices of a Verhulst model are respected in the predicted population size. Length is handled well, for $K = 3$, we have an average length difference of 0.974 between our labels and ground truth, and an average length difference of 2.578 for $K = 1$. With 10 epochs of training per self improvement round, length difference drops to 0 for both K values.

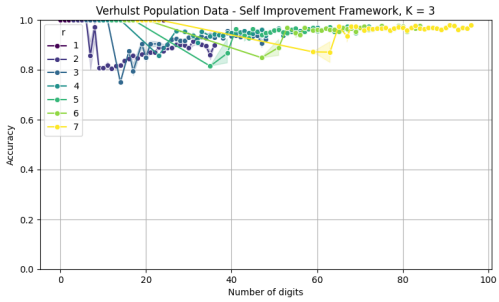


Figure 9: The self improvement framework applied to the Verhulst population data with $K = 3$.

We see that the self improvement framework also works for a Poisson Process. We see large length chains handled accurately by the self improvement framework. By the last round, the average length difference of the labels and the ground truth is 0 when modeling a Poisson Process with a self-improvement trained large language model. The Self Improvement Framework accomplishes significant extrapolation from the initial training distribution. Observations from training data achieve structure recognition and give the correct number of Birth and Death Processes by the end. Table 3 shows that most Birth and Death Processes can achieve perfect structure comprehension by the final self improvement round, and reach a population

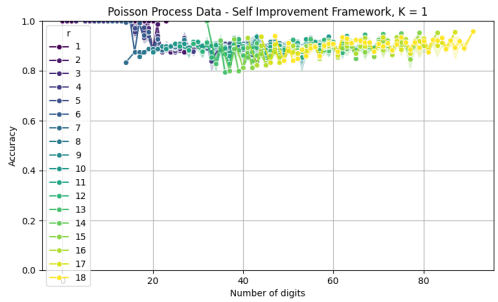


Figure 10: The self improvement framework applied to the Poisson process data with $K = 1$.

size as close as 26.91% to the BirDePy estimate, up to length 37, without access to ground truth beyond the initial distribution of $2 - 7$.

Process	K	Epochs	Max Length	Length Diff	$\Delta\% Z_n$
Verhulst	1	1	30	2.578	77.65%
Verhulst	3	1	28	0.974	38.19%
Verhulst	1	10	40	0	145.19%
Verhulst	3	10	37	0	26.91%
Poisson	1	1	15	0	9.16%
Poisson	3	1	13	0	11.22%
Poisson	1	1	37	0	46.38%
Poisson	3	1	40	0.002	51.16%
Poisson	1	10	40	0.0023	56.04%

Table 3: Metrics for self-improvement trained models on D_n, size 3000; most have perfect length difference, an error that the benchmarks struggle with.

5.5 Parameter Estimation of Self-Improvement Trained Models Through State Models

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

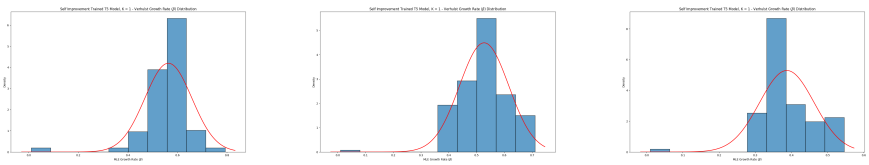


Figure 11: 2-7, 9-11, and 16-21 length β distributions, from left to right. $\hat{\beta} = 0.5875$ (Left), $\hat{\beta} = 0.5050$ (Middle), $\hat{\beta} = 0.3813$ (Right). We take outputs from the self-improvement trained large language model for the Verhulst model, with data generated from $\beta = 0.75$, using appropriate length datasets, and obtain the maximum likelihood parameter estimates to create a distribution.

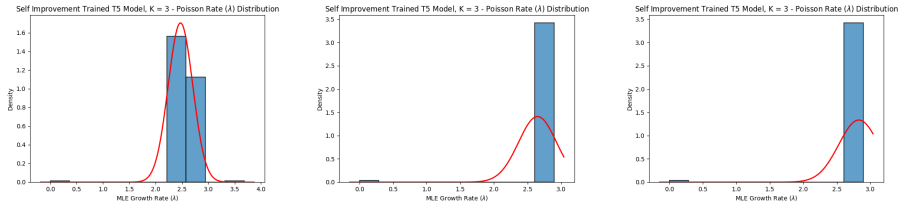


Figure 12: 2-7, 9-11, and 16-21 length λ distributions, from left to right. $\hat{\lambda} = 2.3684$ (Left), $\hat{\lambda} = 2.6316$ (Middle), $\hat{\lambda} = 2.8947$ (Right). We take outputs from the self-improvement trained large language model for the Poisson process, with data generated from $\lambda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

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.

The self-improvement trained model struggles to generalize on test data. While we see long lengths handled precisely when carefully predicting labels from each M_r, a randomly generated test dataset shows that accuracy from the last M_r falls below benchmarks; raw accuracy falls between 27% and 30% on all datasets of the first length bracket for the Verhulst and Poisson.

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. If a Birth and Death Process approximated by one of our large language models estimates a real world system, we can replicate the system and how likely the results of our large language models are given the results of our system. A state model is perfect to this end (Shumway and 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.

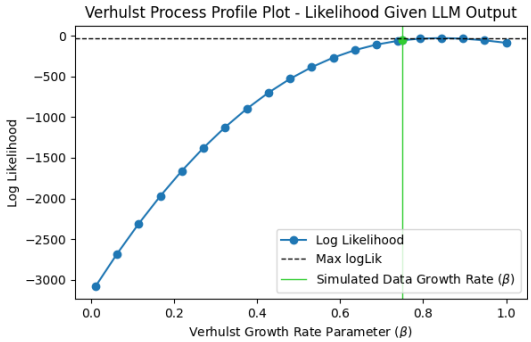


Figure 13: Verhulst Process Profile Plot. The maximum likelihood parameter estimate is 0.84.

Verhulst Population dynamics as implemented inside of a state model measure a Verhulst population large language model. We chain inputs to the T5-small self-improvement trained large language model approximating a Verhulst Birth and Death Process to generate 10 population sizes, starting from the initial state. 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 chain inputs to the T5-small self-improvement trained large language model

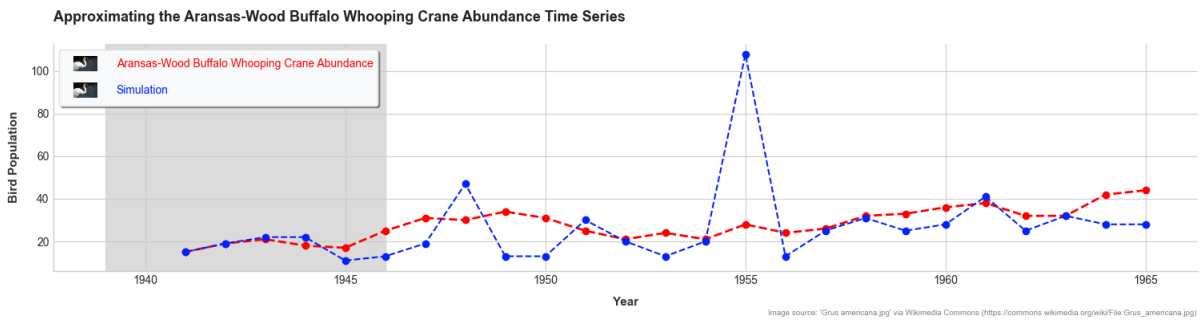


Figure 14: Aransas-Wood Buffalo Whooping Crane abundance simulated beyond an initial training distribution (grey).

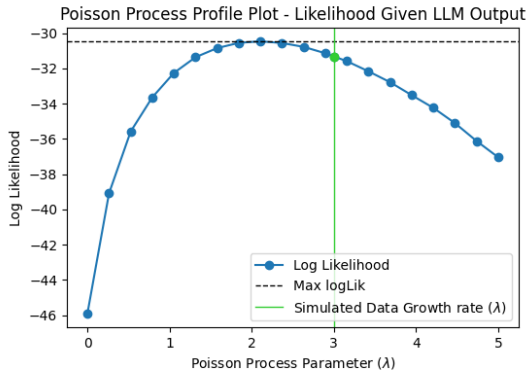


Figure 15: Poisson Process Profile Plot. The maximum likelihood parameter estimate is 2.1.

approximating a Poisson Process and achieve similar success. The Profile Plot shows that the maximum likelihood growth parameter rate is consistent with the parameter rate used to generate data, showing that $\theta = \{\lambda\} = \{3\}$ has a high likelihood when used with the state model, and is not far off from the maximum likelihood estimator $\hat{\theta} = \{\hat{\lambda}\} = \{2.1\}$.

6 Application to Bird Populations

We train a T5-small model on Verhulst Population data with $\beta = 0.35, \nu = 0.3$. Figure 14 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.

7 Limitations

We extend Birth and Death Processes from lengths of 2-7 up to 40. Given adequate hardware, it is reasonable to extrapolate from our findings that the self improvement framework can be extended even further; we see no decrease in quality at length 40 from the self improvement framework. Our findings are limited to transition probabilities from the Verhulst model and a Poisson model; future studies may study how large the initial training distribution needs to be to learn appropriate transition probabilities. We are also limited by the specifications of the NVIDIA A40 GPU.

8 Conclusion

Birth and Death Processes are modeled well by large language models. Length generalization remains a difficult problem; our benchmarks show that accuracy and errors in length plague larger length brackets, yielding poor results. Given hard labels at these lengths, fine tuning showed the most success at solving these problems. Training at higher length brackets makes our benchmarks more successful. The Self Improvement Framework allows us to extend large language models approximating Birth and Death Processes to lengths well outside the training distribution without hard labels. 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. While outputs from benchmarks have lengths off by 14 or 15 in the highest length bracket, self-improvement trained models achieve perfect length differences. We conclude that, although the self improvement framework is the only solution that does not require ground truth labels, fine tuning, promping, and the self improvement framework are all solutions that can be applied to model Birth and Death Processes and handle longer lengths.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

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

Anindya Bhadra, Edward L Ionides, Karina Laneri, Mercedes Pascual, Menno Bouma, and Ramesh C Dhiman. 2011. 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.

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

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

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

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

Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, and 1 others. 2013. *An introduction to statistical learning: with applications in R*, volume 103. Springer.

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

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

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

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

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

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

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

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

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

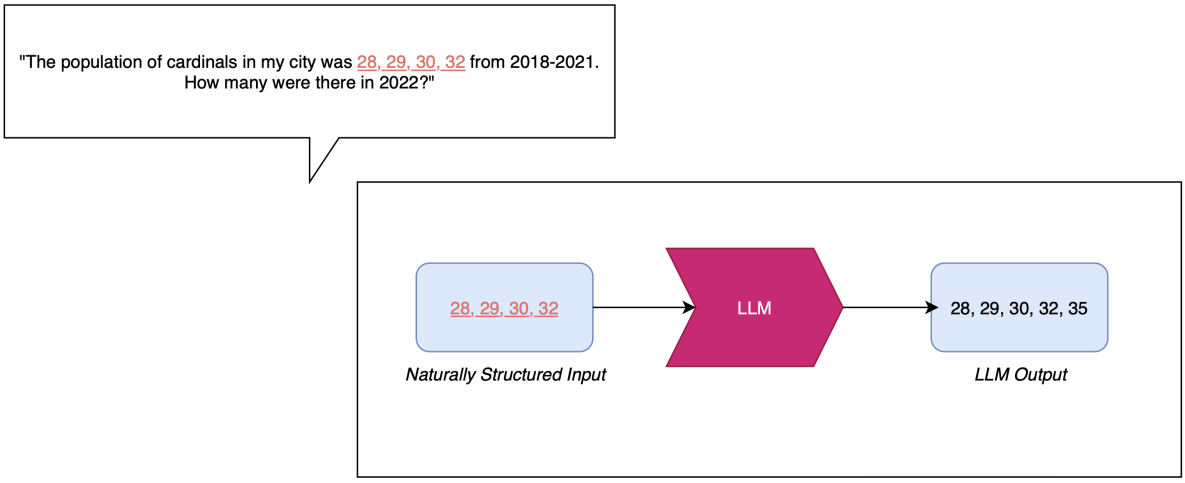
William J Thompson. 2001. Poisson distributions. *Computing in Science & Engineering*, 3(3):78–82.

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

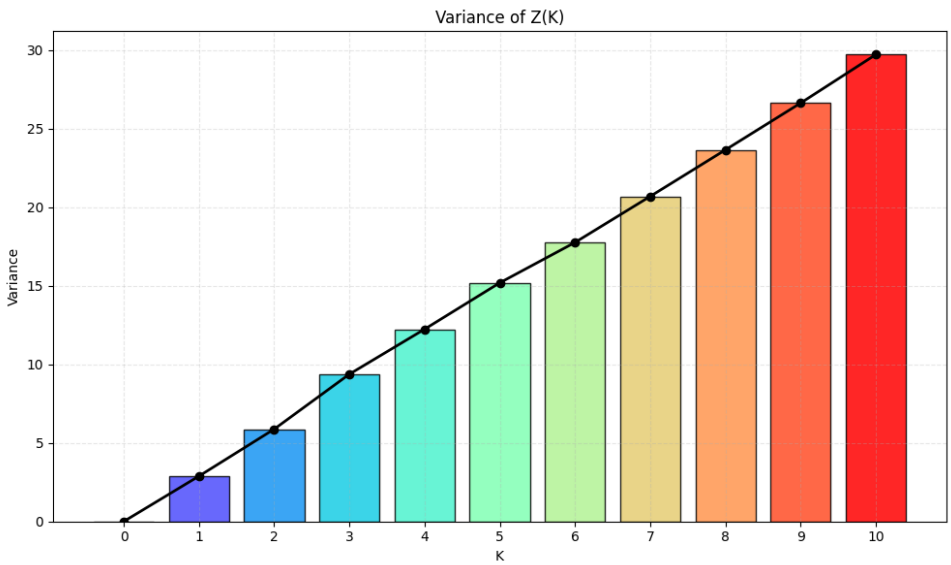
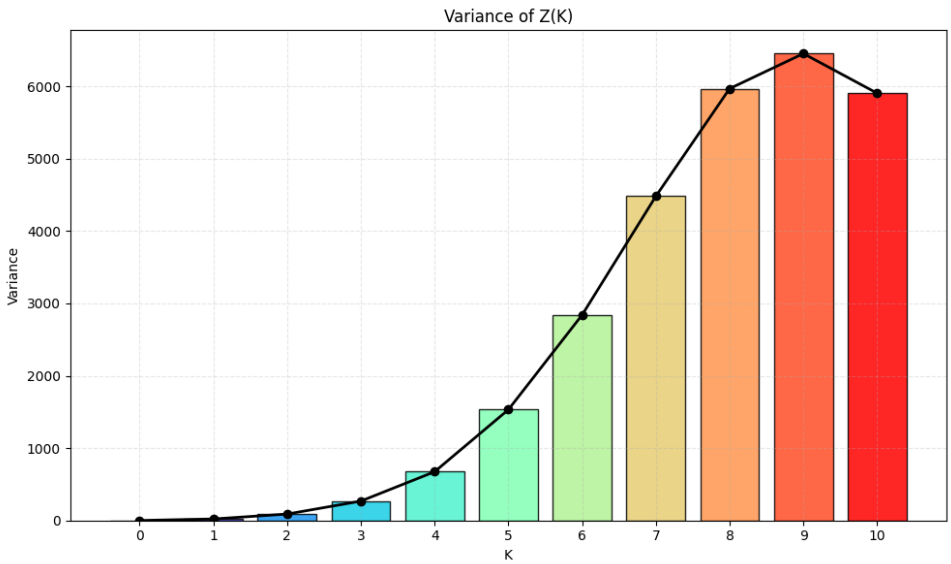
A LLM Disclosure

LLM's have been used in this project to conduct help collecting papers during literature review, locate available data, debug and assist during coding, and assist in latex and visualization formatting.

B Conversational Agent Diagram



C Poisson Process Variance by K



D Verhulst Process Variance by K