How does Multi-Task Training Affect Transformer In-Context Capabilities? Investigations with Function Classes

Anonymous ACL submission

Abstract

Large language models (LLM) have recently shown the extraordinary ability to perform unseen tasks based on few-shot examples provided as text, also known as in-context learning (ICL). While recent work has attempted to understand the mechanisms driving ICL, few have explored training strategies that incentivize these models to generalize to multiple tasks. Multi-task learning (MTL) for generalist models is a promising direction that offers transfer learning potential, enabling large parameterized models to be trained from simpler, related tasks. In this work, we investigate the combination of MTL with ICL to build models that efficiently learn tasks while being robust to out-of-distribution examples. We propose several effective curriculum learning strategies that allow ICL models to achieve higher data efficiency and more stable convergence. Our experiments [1] reveal that ICL models can effectively learn difficult tasks by training on progressively harder tasks while mixing in prior tasks, denoted as mixed curriculum in this work.

1 Introduction

Recently, the emergence of in-context-learning capabilities in LLMs has revolutionized the field of NLP (Wei et al., 2022a). By pre-training with next-word predictions, these models can be prompted with few-shot examples and make accurate in-context predictions during inference (Brown et al., 2020). The ICL capability demonstrated even by smaller Transformer models presents an alternative way to understand LLMs (Dong et al., 2023; Li et al., 2023a; Lu et al., 2023). To empirically understand this phenomenon, Garg et al. (2022) focus on learning a single function class in-context by a Transformer model. Their model achieves competitive normalized mean-squared error (MSE) compared to the optimal ordinary least squares estimator when performing in-context linear regression. Nevertheless, these models sometimes fail to converge and often struggle to generalize to more challenging function classes. While the follow-up studies (Akyürek et al., 2023; Von Oswald et al., 2023; Yang et al., 2023) have extensively analyzed how these models conduct ICL, little work exists exploring how training on *multiple function classes* can enable transformer models to generalize and perform ICL more efficiently. As we believe that these generalist models are typically designed to perform multiple tasks, there is a pressing need to study the multi-task ICL capability of these models, which is still missing in the literature.

From prior multi-task training (MTR) studies (Zhang et al., 2023; Ruder, 2017; Weiss et al., 2016), models can be trained on multiple related tasks to improve their performance on individual tasks. Despite its popularity, multi-task learning has been difficult to understand in Transformer models when trained on natural language, most likely due to the difficulty of ranking and scheduling language tasks (Crawshaw, 2020). However, the newly introduced framework of learning function classes in context (Garg et al., 2022) provides an easier way to study this multi-task training paradigm. For example, for a polynomial function class, its difficulty can be scaled by changing its degree (e.g., linear to quadratic), or changing the input distribution (e.g., standard Gaussian to Gaussian distribution with decaying eigenvalues). This allows us to understand the ICL capabilities to transfer across similar function classes from multi-task training. Motivated by this, we conduct a systematic analysis by training a Transformer on varying function class families and input distributions in a multi-task manner to examine if the same principles from MTR carry over into ICL.

During training, we explore different curriculum learning strategies to schedule the ICL tasks of multiple function classes: (*mixed*, *sequential*, *random*)

[1] Code and models will be released on acceptance

(§2.3). For benchmarking, we train another set of models only on a single function class family following (Garg et al., 2022). We quantitatively and qualitatively compare our models trained with and without curriculum across all tasks and analyze the normalized mean-squared error (MSE) and attention matrices (§3). Our experiments show that curriculum learning is more data-efficient, achieving comparable performance to single-task models using only 1/9 of the training data. These curriculum models can also obtain an optimal MSE in function classes where none of the single-task models converge.

2 Methods

2.1 Problem Definition

Following Garg et al. (2022), we define the problem of ICL as passing in an n-shot sequence $(x_1, f(x_1), x_2, f(x_2), \ldots, x_n, f(x_n), x_{n+1})$ to the Transformer and generating an output for $f(x_{n+1})$, where the examples have not been seen during training. We refer to this n-shot prediction problem, where input is given in pairs, as in-context learning.

We consider a data-generating process where d-dimensional covariates are drawn $x_i \sim \mathcal{D}_x$ and a function $f \sim \mathcal{F}$ where $\mathcal{D}_x$ is any arbitrary distribution and $\mathcal{F}$ is the class of functions related to single-index normalized Hermite polynomials.

2.2 Tasks

We explore two types of separate tasks: learning a function class and learning a data distribution (in Appendix). We consider a single-index function:

$$f(x) = \varphi(\langle x, w \rangle).$$

Function Class Learning We look at the class of functions derived from normalized probabilist's Hermite polynomial, $\frac{1}{\sqrt{n!}} He_n(x)$ which satisfies orthogonality. This is useful as it guarantees that the function values of all tasks are uncorrelated. For each task, we sample w uniformly from the unit sphere. We define $K = 3$ polynomial function classes as follows: denoting $t = \langle x, w \rangle$, we pick $\varphi \in \{\varphi_{\text{linear}}, \varphi_{\text{quadratic}}, \varphi_{\text{cubic}}\}$ for three function classes $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3$ respectively.

$$\varphi_{\text{linear}}(t) = t,$$

$$\varphi_{\text{quadratic}}(t) = \frac{1}{\sqrt{2}}(t + \frac{1}{\sqrt{2}}(t^2 - 1))$$

$$\varphi_{\text{cubic}}(t) = \frac{1}{\sqrt{3}}(t + \frac{1}{\sqrt{2}}(t^2 - 1) + \frac{1}{\sqrt{6}}(t^3 - 3t))$$

2.3 Curriculum Learning

We define the total training steps T to be 500,000, where the t-th training step ranges from $t = 1, 2, \ldots, T$. Our curriculum learning strategy (Sequential, Mixed, Random) is used to allocate our K tasks across training time. In this paper, we explore $K = 3$ function classes defined earlier.

Sequential Curriculum Given T total training steps, we split the training steps into K partitions. Within the k-th partition of training steps, we train the model on learning a function from the k-th function class, in order of increasing difficulty:

$$f \sim \begin{cases} \mathcal{F}_1 & 0 \le t \le \frac{T}{3} \\ \mathcal{F}_2 & \frac{n}{3} \le t \le \frac{2T}{3} \\ \mathcal{F}_3 & \frac{2T}{3} \le t \end{cases}$$

Mixed Curriculum Given T total training steps, we again split the training steps into K partitions. Let ξ be (uniformly) drawn from $\{1, 2\}$ and ζ be (uniformly) drawn from $\{1, 2, 3\}$. We select tasks from the previous k partitions with equal probability ($\mathbf{1}$ denotes the indicator function):

$$f \sim \begin{cases} \mathcal{F}_1 & 0 \le t \le \frac{T}{3} \\ \sum_{s=1}^{2} \mathbf{1}(\xi = s)\mathcal{F}_s & \frac{T}{3} \le t \le \frac{2T}{3} \\ \sum_{s=1}^{3} \mathbf{1}(\zeta = s)\mathcal{F}_s & \frac{2n}{3} \le t \end{cases}$$

Random Curriculum At each training step t, randomly sample from the list of K tasks with distribution equal probability:

$$f \sim \sum_{s=1}^{3} \mathbf{1}(\zeta = s)\mathcal{F}_s, \quad 0 \le t \le T$$

2.4 Attention Analysis

To understand how single and multi-task models learn, we analyze the Transformer's self-attention weights. Specifically, we mask out the attention matrices for each head to filter out the self-attention scores between each $f(x_i)$ token and its corresponding x_i token. To summarize the head's inclination to attend to previous tokens, we aggregate these scores by taking the mean across all $f(x_i)$ tokens. We then do this for all attention heads in all layers and plot this as a head-by-layer heatmap. We define a "retrospective head" as an attention head that has a lighter value in the heatmap, indicating that this specific head learns to attend to the previous input token when constructing a representation for the current token, a natural pattern that encourages understanding of the input pairs.

3 Results

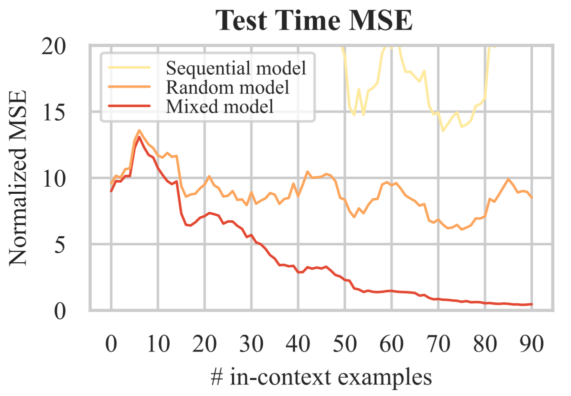

Figure 1: Comparison of moving average of all three curriculum learning strategies when evaluated on quadratic function class dataset during test time. The mixed curriculum is the only model that is able to achieve an accurate normalized MSE. The random curriculum performs comparatively worse, whereas the sequential curriculum performs substantially worse (y-axis is limited in order for mixed and random curricula to be differentiated).

Curriculum Strategy Comparison Figure 1 shows that the mixed curriculum outperforms both the random and sequential curriculum when evaluating all models on a quadratic function class dataset during test time. We find that the mixed curriculum strategy provides the most benefit towards learning multiple tasks. This is further validated in Supplementary Figure 5, which shows that the mixed curriculum is most stable over all tasks, achieving an accurate solution after sufficient few-shot examples (20/80/90-shot examples for Linear/Quadratic/Cubic respectively). We hypothesize this is due to stable periods of training, where the model is able to adapt to the given function class, whereas the random curriculum does not have such a schedule. Additionally, mixed curriculum likely outperforms sequential curriculum because including tasks from previous training blocks mitigates catastrophic forgetting (Zhai et al., 2023). Therefore, we stick with the mixed curriculum model in the following experiments.

Qualitative Attention Analysis Figure 2 displays how masking 7 retrospective heads (as defined in §2.4) causes a significant increase in normalized MSE compared to 7 non-retrospective heads in the mixed curriculum model. Using our attention analysis in Supplementary Figure 4, we identify retrospective heads as those with yellow values, whereas non-retrospective heads are highlighted with dark purple values. This supports the theory that specific heads may be reasonable for the ICL abilities of these models (Olsson et al., 2022). Additionally, we find that the same attention heads have high scores across related tasks, indicating that these models are conducting approximations, rather than learning the true tasks.

Curriculum Learning Convergence Figure 3 reveals 60% of mixed curriculum models converge, whereas 0% of the single-task models trained on quadratic function classes converge. Specifically, these models do not achieve optimal (below 1) normalized MSE during training time and at test time. We believe curriculum learning aids in this task, as we allow the model to warm up the training with the objective (calculate $f(x)$ from x) on easier tasks. In contrast, the poor performance of the single-task models may be explained by their cryptic attention patterns (Supp Fig. 2). These findings help us understand how curriculum learning can be used to learn difficult function classes that are otherwise unlearable by single-task models.

Curriculum Learning Data Efficiency Figure 4 illustrates the performance of a single-task model and a mixed curriculum model during training when evaluated on a cubic function class validation dataset. Our experiments uncover that the mixed curriculum model can improve data efficiency, learning harder tasks with fewer examples. The mixed curriculum model is pre-trained on 1/9 of the training examples seen by the single-task cubic model, yet the mixed curriculum model has better performance on the validation set. Pulling from qualitative attention analysis, we hypothesize that the mixed curriculum model is able to use its approximate understanding of the linear and quadratic function classes to improve the initial normalized MSE of a cubic function class. This explains why the cubic model starts at 450 normalized MSE, whereas the mixed model starts at 200 normalized MSE. When analyzing both models at test time (Supp Fig. 3, 5) the mixed model has comparable performance to the single-task cubic model. These findings suggest that curriculum learning can assist data efficiency by making use of transfer learning from easier tasks.

4 Discussion

In this paper, we examine how different curriculum learning strategies affect Transformer's in-context learning capability. We first introduce the three

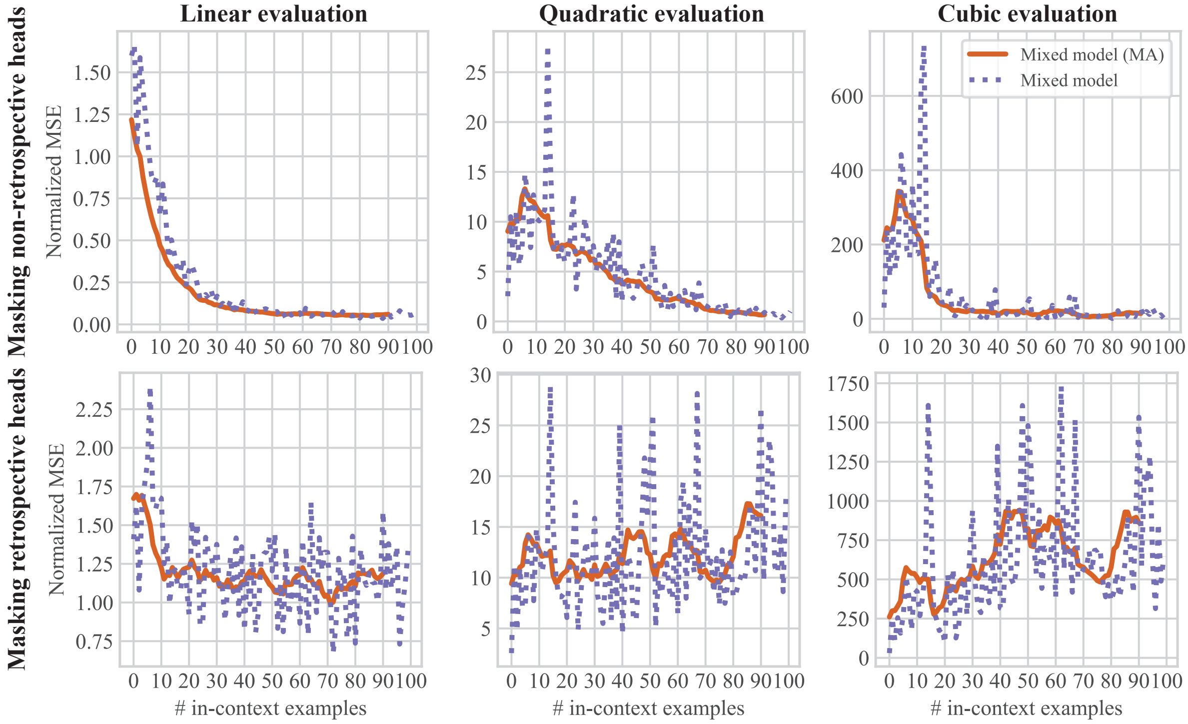

Figure 2: Masking retrospective heads *(bottom row)* causes significant increase in normalized MSE compared to non-retrospective heads *(top row)* in the mixed curriculum model.

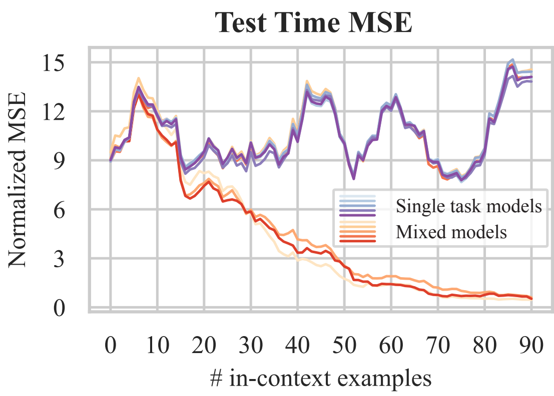

Figure 3: Comparison of the moving average of five different seeded single-task (*blue-purple*) and curriculum models (*orange-red*) evaluated on a quadratic function class dataset during test time. Mixed models are able to learn quadratic function classes whereas the single task models are unable to, indicated by the spikes and upward trend in normalized MSE.

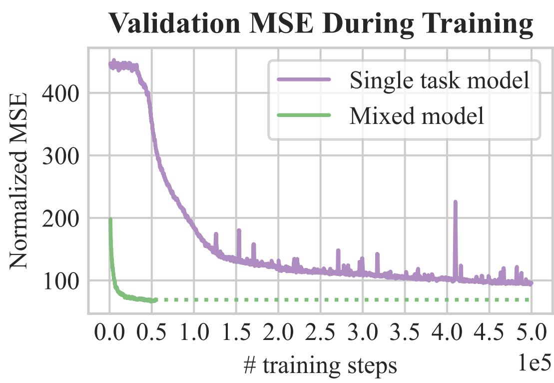

Figure 4: Comparison of moving average normalized MSE of a single-task cubic model to a mixed curriculum model during training. Data points are generated by evaluating the model on a separate validation set of cubic function examples. The mixed curriculum model is initialized with a checkpoint trained on linear and quadratic function examples, while the single-task model is initialized with random weights.

types of curriculum (mixed, random, sequential) along with the two tasks (function class learning and distribution learning). We then compare these curriculum models against models that we only train on a single specific task and evaluate them across related tasks. This reveals that the mixed curriculum provides the best results, as well as increases data efficiency and model convergence. Through our analysis of attention, we show that these multi-task models had high attention scores across related tasks in the same heads, and that if we mask these heads during test time, the accuracy of these models drop drastically, indicating that specific heads are responsible for ICL. These results provide an important insight into how we can better pre-train LLMs to in-context-learn efficiently.

Limitations

Our work investigates ICL on standard function classes which can be mathematically defined, however it may be difficult to extend our work to natural language tasks as they are hard to define. The extensibility of our work to natural language tasks therefore remains an open question. We make use of three well-known scheduling methods, however, more effective curriculum learning scheduling strategies should be investigated. We work with a relatively small model, thus our results may not be transferable to larger models such as Llama-2 or GPT-4 and we work with noiseless data which may inflate the accuracy. Lastly, we acknowledge that in-context learning can be inconsistent (models only learn approximations for tasks and have varying performance across seeds) and should not be used in high-risk situations.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2023. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Center for High Throughput Computing. 2006. Center for high throughput computing.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A survey on in-context learning.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. In *Advances in Neural Information Processing Systems*, volume 35, pages 30583–30598. Curran Associates, Inc.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. 2023a. Transformers as algorithms: Generalization and stability in in-context learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19565–19594. PMLR.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. 2023b. Transformers as algorithms: Generalization and stability in in-context learning. In *International Conference on Machine Learning*, pages 19565–19594. PMLR.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Computing Surveys*, 55.

Sheng Lu, Irina Bigoulaeva, Rachneet Sachdeva, Harish Tayyar Madabushi, and Iryna Gurevych. 2023. Are emergent abilities in large language models just in-context learning?

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, Seattle, United States. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2022. In-context learning and induction heads. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy. Association for Computational Linguistics.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35151–35174. PMLR.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. 2023. Larger language models do in-context learning differently.

Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big Data*, 3(1):9.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Steve Yadlowsky, Lyric Doshi, and Nilesh Tripuraneni. 2023. Pretraining data mixtures enable narrow model selection capabilities in transformer models.

Jiaxi Yang, Binyuan Hui, Min Yang, Binhua Li, Fei Huang, and Yongbin Li. 2023. Iterative forward tuning boosts in-context learning in language models.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Fan Yin, Jesse Vig, Philippe Laban, Shafiq Joty, Caiming Xiong, and Chien-Sheng Wu. 2023. Did you read the instructions? rethinking the effectiveness of task definitions in instruction learning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3063–3079, Toronto, Canada. Association for Computational Linguistics.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2023. Investigating the catastrophic forgetting in multimodal large language models.

Zhihan Zhang, Wenhao Yu, Mengxia Yu, Zhichun Guo, and Meng Jiang. 2023. A Survey of Multi-task Learning in Natural Language Processing: Regarding Task Relatedness and Training Methods. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*.

Appendix

A Experimental Settings

We train on the GPT-2 model (22.4 million parameters) with 12 heads, 8 layers, and an embedding size of 256 over 500,000 steps, where each batch size is 64. Each batch consists of 100 $(x_i, f(x_i))$ pairs (we expect higher order polynomials to require more in-context examples to converge). During training, we evaluate each model every 2,000 steps on a validation set with size of 32,000. During test time evaluation, we evaluate each model on 64 randomly selected examples. We train GPT-2 using a A100-SXM4-80GB provided by the Center for High Throughput Computing (2006).

B Distribution Learning

In addition to different function classes, we explore training data generated from different distributions, given that recent literature has shown that these models do not perform well under distributional shifts (Garg et al., 2022; Yadlowsky et al., 2023). Particularly, we sample inputs x_i from (i) Gaussian distributions, (ii) skewed Gaussian distributions (decaying eigenvalues), and (iii) student-t distributions (df=4). Attention matrices (Supp Fig. 6 and 8) and normalized MSE (Supp Fig. 7 and 9) across tasks may be found for both single-task and curriculum-based models in the Supplementary Materials.

C Instruction Prompting

We explore two sets of instruction prompting architectures: one-hot encoded vectors and preset instruction vectors. The goal of instruction prompting was to evaluate whether our objective could benefit from instruction prompting the way language translation or other NLP tasks do.

C.1 One Hot Encoded Vectors (OHEI)

After generating our $(x_i, f(x_i))$ pairs, we append a single one hot encoded vector, p, to the beginning of the sequence, with the one hot encoding corresponds to the "task":

$$p = \begin{cases} p_0 = 1 & \varphi = \varphi_1 \\ p_1 = 1 & \varphi = \varphi_2 \\ p_2 = 1 & \varphi = \varphi_3 \end{cases}$$

We then apply a linear transformation to transform the concatenation into the dimension of our transformer, 256.

C.2 Preset Instruction Vectors (PI)

After we use a linear transformation to transform our $(x_i, f(x_i))$ pairs to the input dimension of our transformer,256, we append a unique vector, $p \sim \mathcal{N}(0, I_d)$, that has been sampled from an isotropic Gaussian distribution. This "instruction vector" remains constant throughout the training of all models, but remains different for each of the different tasks.

C.3 Instruction prompting remains unclear

Supplementary figure 1 shows the comparison of a mixed curriculum model with no instruction prompting, to the two instruction prompting architectures listed above, evaluated over all function class tasks. Applications of the one hot encoded instruction (OHEI) vector to the mixed model causes minimal improvement, whereas application of the preset instruction (PI) vector to the mixed model worsens model performance in the quadratic and cubic function class evaluation during test time. We believe the former has minimal effect in performance as the one-hot encoded vectors may just be seen as noise, whereas the latter most likely worsens the ability of the model to learn the task as it may be seen as an extreme version of noise (i.e. it disrupts the flow of $x_i, f(x_i)$ confusing the model). Overall, we believe that instruction tokens may not be tractable in this setting due to the difficult of learning a 20-dimensional instruction.

D Related Work

In-context Learning In-context learning has been around for a few years now (Dong et al., 2023), and many papers have analyzed in-context learning with natural language (Min et al., 2022b; Xie et al., 2022; Min et al., 2022a). It wasn't until Garg et al. (2022) that papers started analyzing ICL through the paradigm of function class learning. Garg et al. (2022) released a paper that showed that transformers could learn linear regression close to the OLS estimator, and other more complex function classes with respectable accuracy. However, they found that some function classes were hard to learn or did not converge (e.g. skewed gaussian). Yadlowsky et al. (2023) looked at a framework similar to Garg et al. (2022), where they explored training models on a mixture of function classes however they do not explore curriculum strategies. Many papers also explore how in-context learning works, with current literature pointing to it being a fuzzy gradient descent, (Akyürek et al., 2023; Von Oswald et al., 2023; Yang et al., 2023). Additional theoretical work studies how transformers can implement near-optimal regression algorithms and describe stability conditions for ICL (Li et al., 2023b).

Attention Analysis Until Vaswani et al. (2017), attention was not widely used in neural networks, however Transformers have revolutionized our capabilities of performing tasks in a variety of fields. Given the power behind attention, we wanted to figure out a way to analyze it, similar to what has been done in previous work (Clark et al., 2019). (Olsson et al., 2022; Elhage et al., 2021) found that specific heads, specified as "induction heads", were responsible for the in-context-learning ability of transformers, both in large and small transformers. To measure this, they created their own metric. Interested in seeing if specific heads attended to specific tasks in a multi-task framework, we decided to visualize the attention matrix. (Vig and Belinkov, 2019) showed a simple and easy way to visualize attention, that was interpretable. We used this as a proxy to develop our own analyses of the attention matrices. Other recent work focuses on summarizing attention flow through transformer models from input embeddings to later layers with attention rollout (Abnar and Zuidema, 2020).

Instruction prompting Prompting has been widely used in natural language tasks to improve accuracy and tends to be robust to variations during test time (Liu et al., 2023). Wei et al. (2023) showed that models of different architectures responded differently to instruction tokens, with the formatting of the instruction effecting multi-task settings. Yin et al. (2023) showed that providing key information in tasks in a common format improved the ability of the model to learn the task. Recently frameworks have emerged which prompt LLMs with intermediate reasoning steps to elicit better reasoning capabilities (Wei et al., 2022b), known as Chain of Thought (CoT) prompting. (Besta et al., 2023) and (Yao et al., 2023) extend CoT prompting to consider multiple reasoning paths to improve performance. Future work may consider using these methods to improve in-context learning in the multi-task setting.

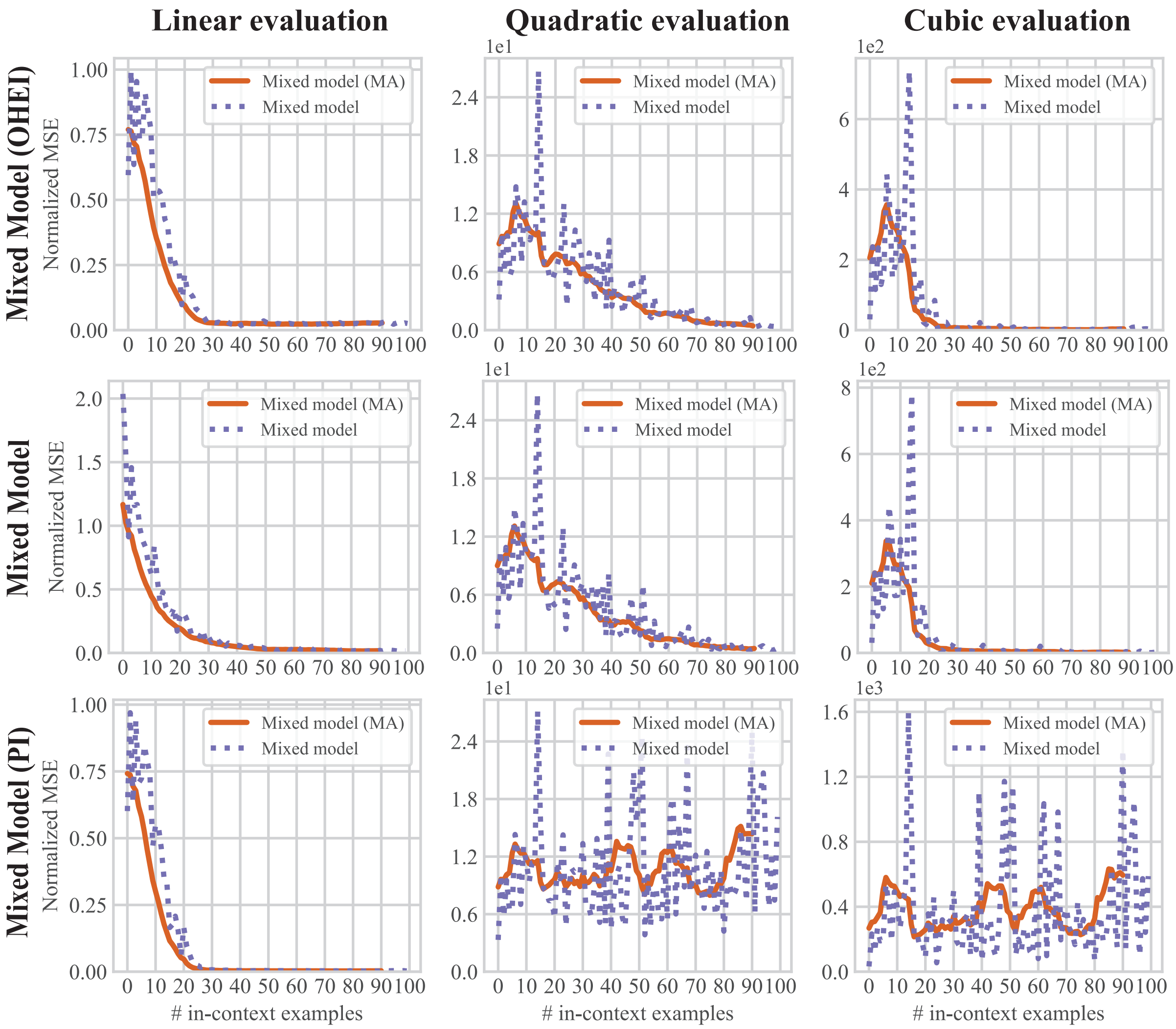

Figure 1: Normalized MSE over number of in-context examples for mixed curriculum model, mixed curriculum model with one hot encoded instruction (OHEI) vector and mixed curriculum model with preset instruction (PI) vector. Solid line represents the moving average (window=10) whereas the dashed line is the actual value. Scientific notation is used for the y-axis.

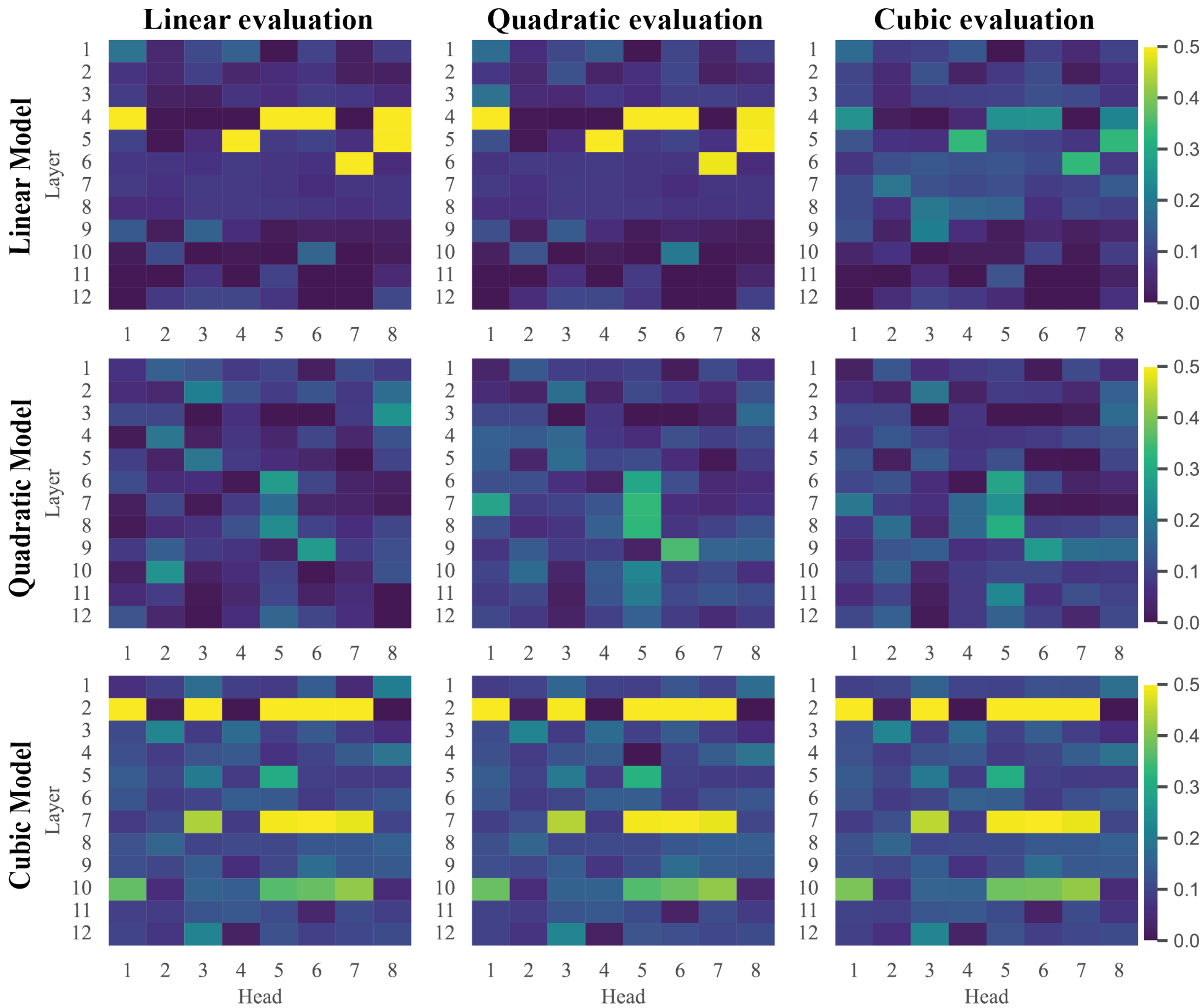

Figure 2: Attention analysis for single-task function learning models.

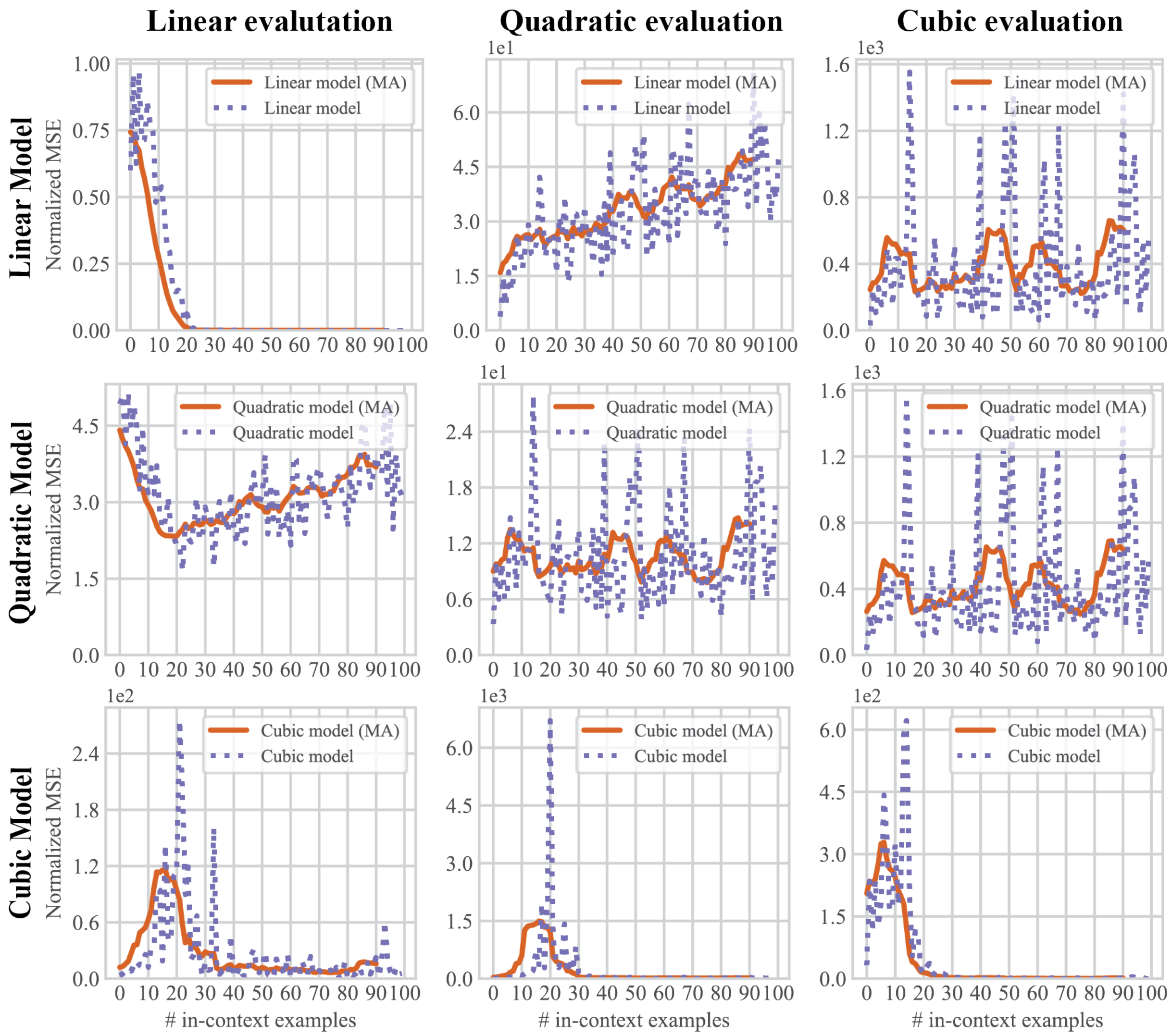

Figure 3: Normalized MSE over number of in-context examples for single-task function learning models. Solid line represents the moving average (window=10) whereas the dashed line is the actual value. Scientific notation is used for the y-axis.

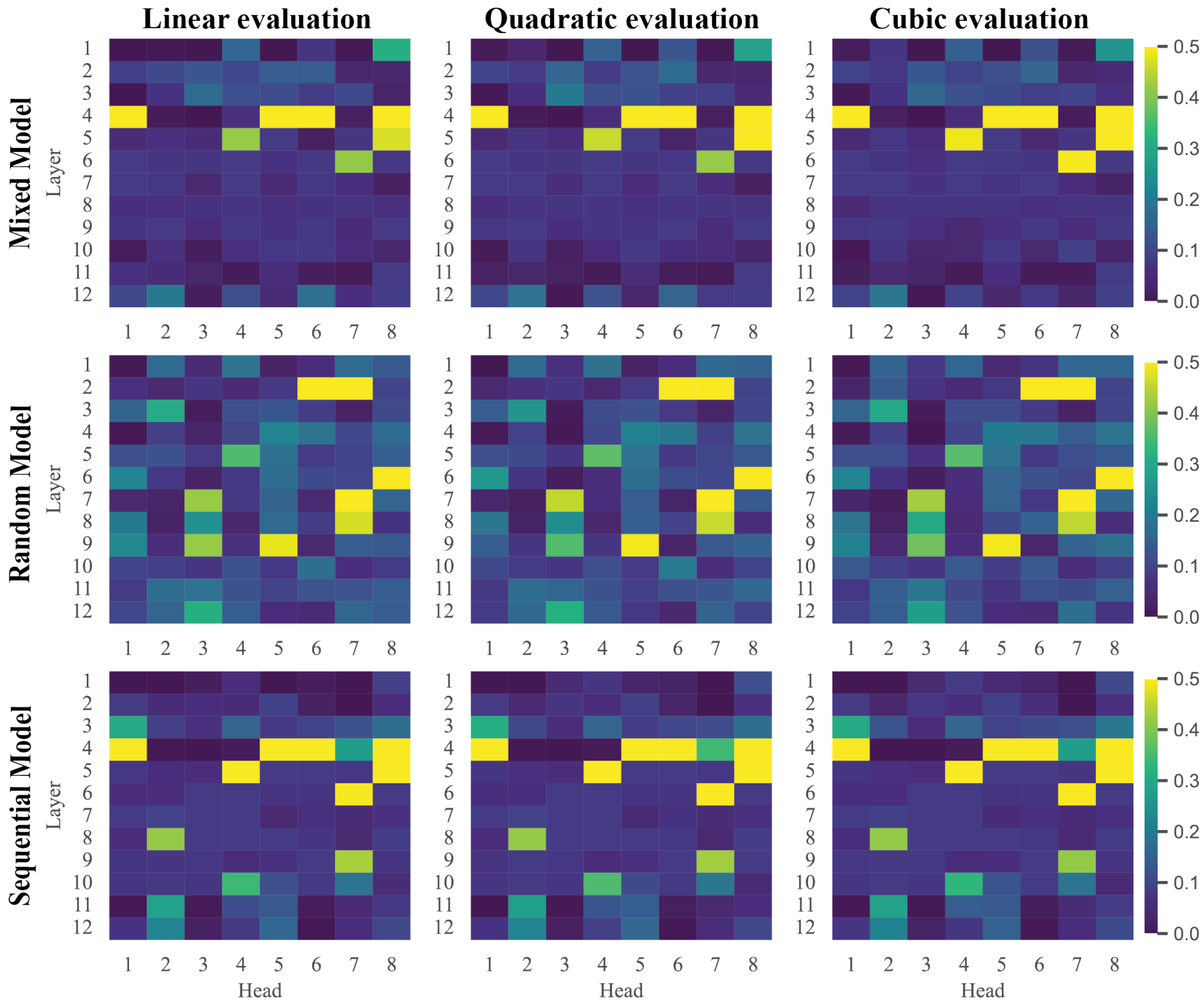

Figure 4: Attention analysis for curriculum function learning model.

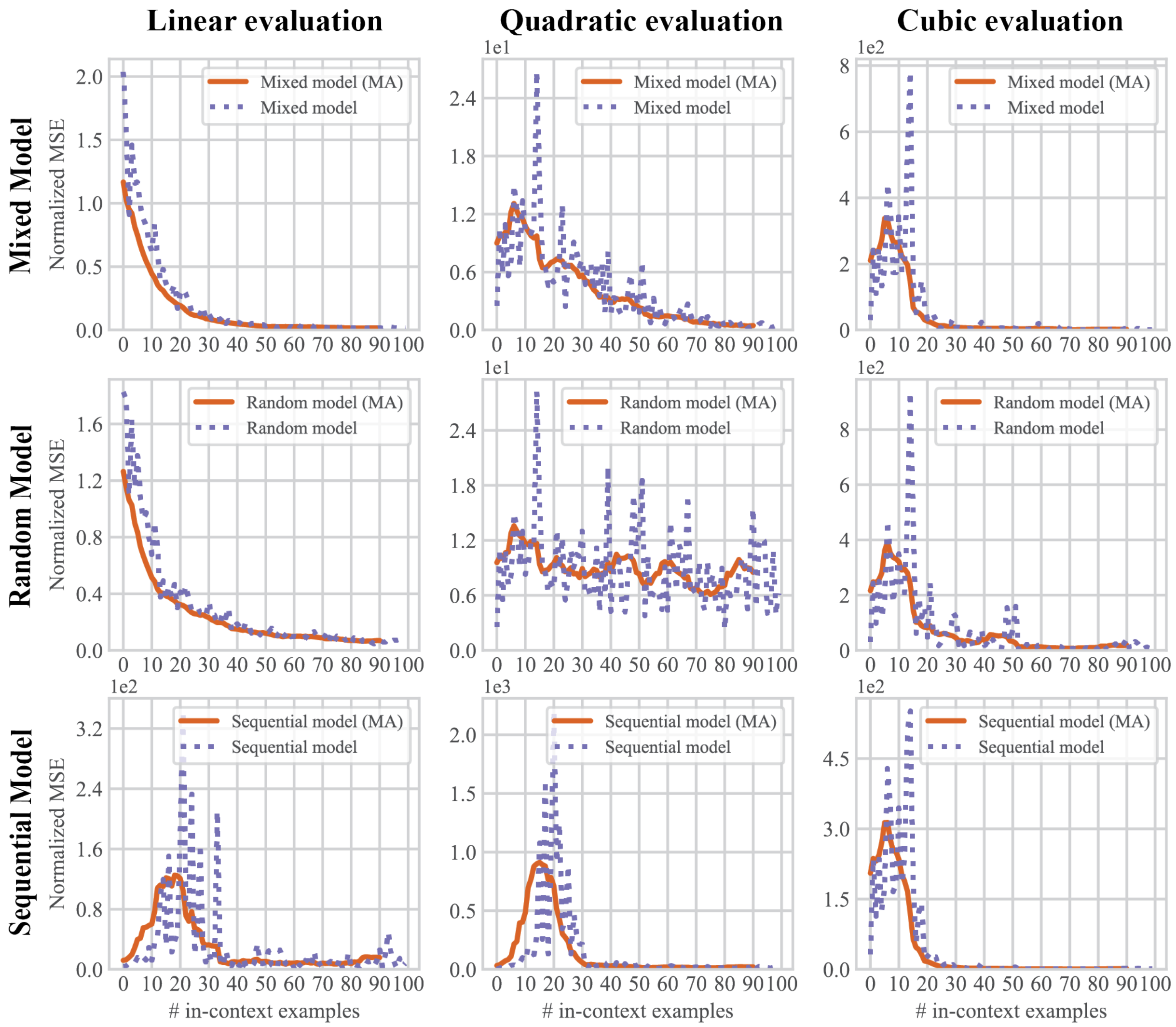

Figure 5: Normalized MSE over number of in-context examples for curriculum function learning models. Solid line represents the moving average (window = 10) whereas the dashed line is the actual value. Scientific notation is used for the y-axis.

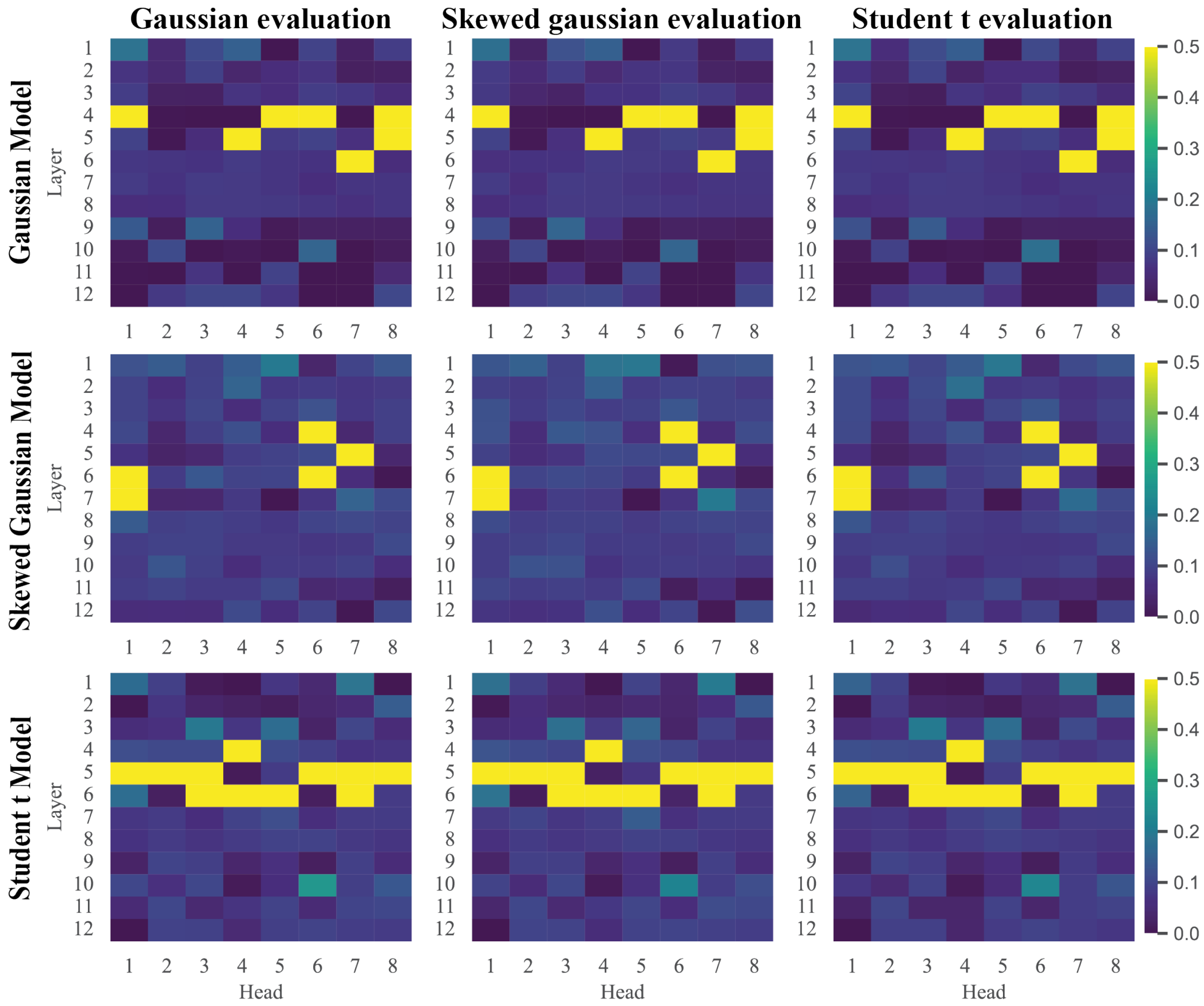

Figure 6: Attention analysis for single-task distribution learning models.

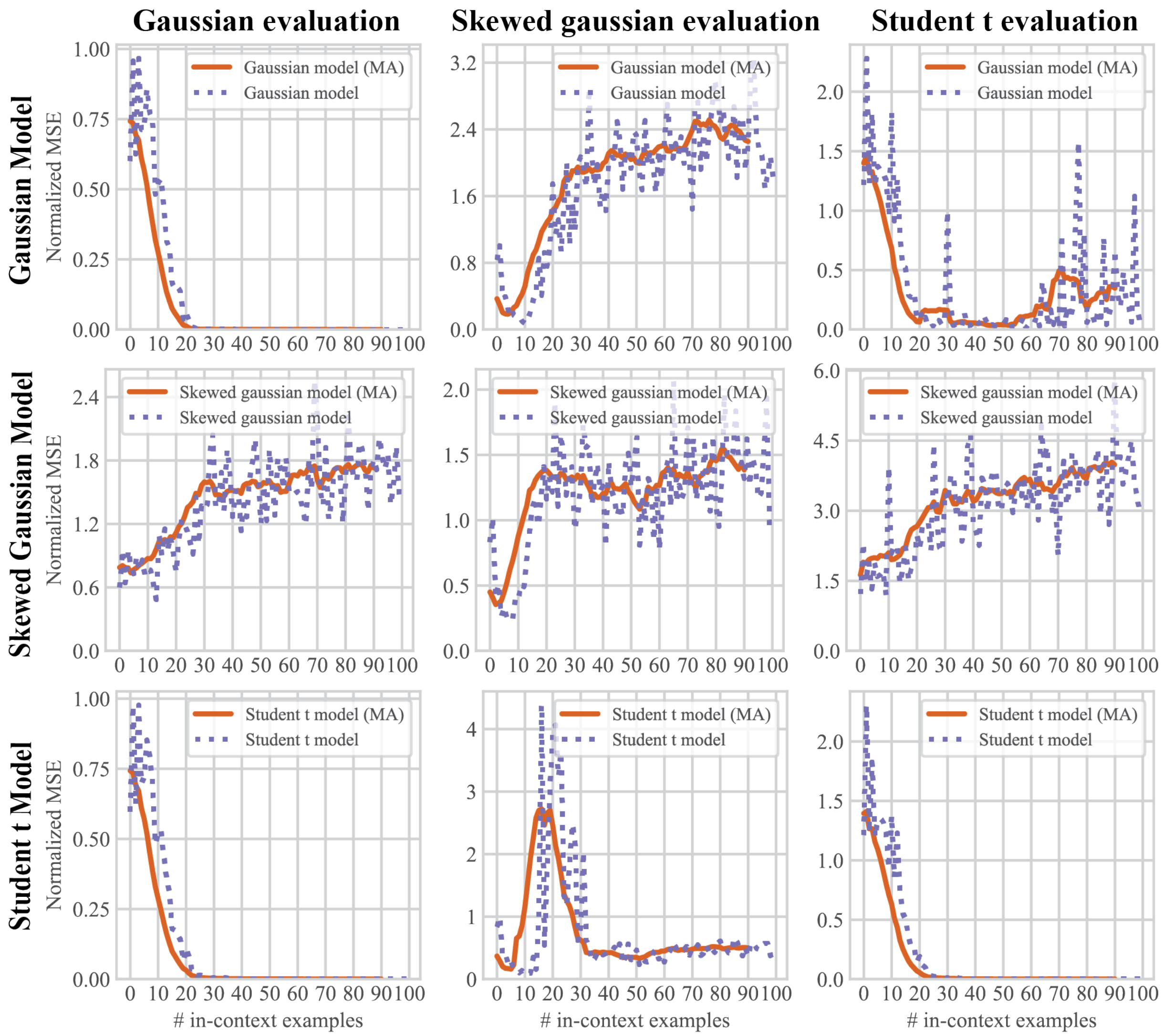

Figure 7: Normalized MSE over number of in-context examples for single-task distribution learning models. Solid line represents the moving average (window = 10) whereas the dashed line is the actual value. Scientific notation is used for the y-axis.

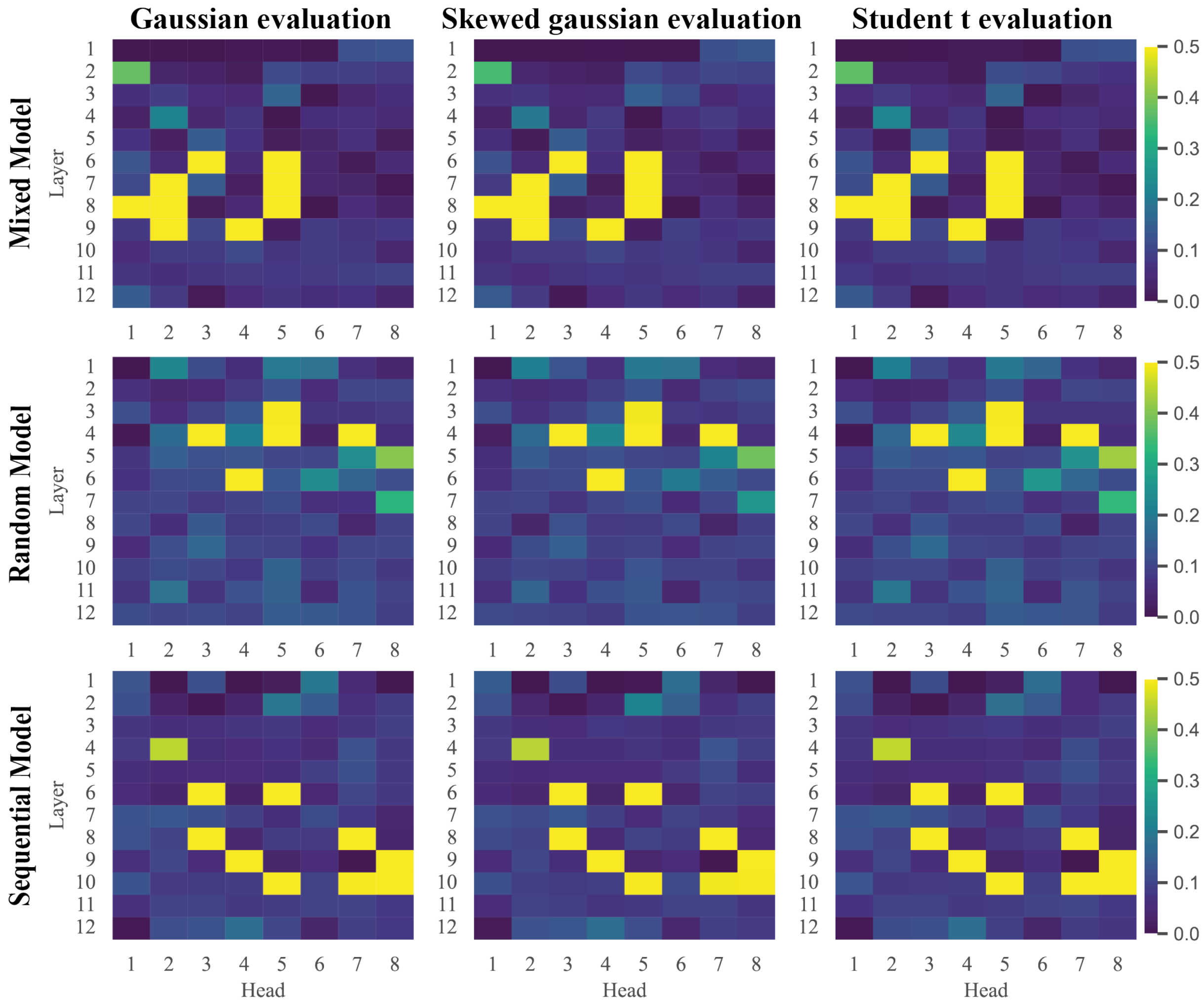

Figure 8: Attention analysis for curriculum distribution learning.

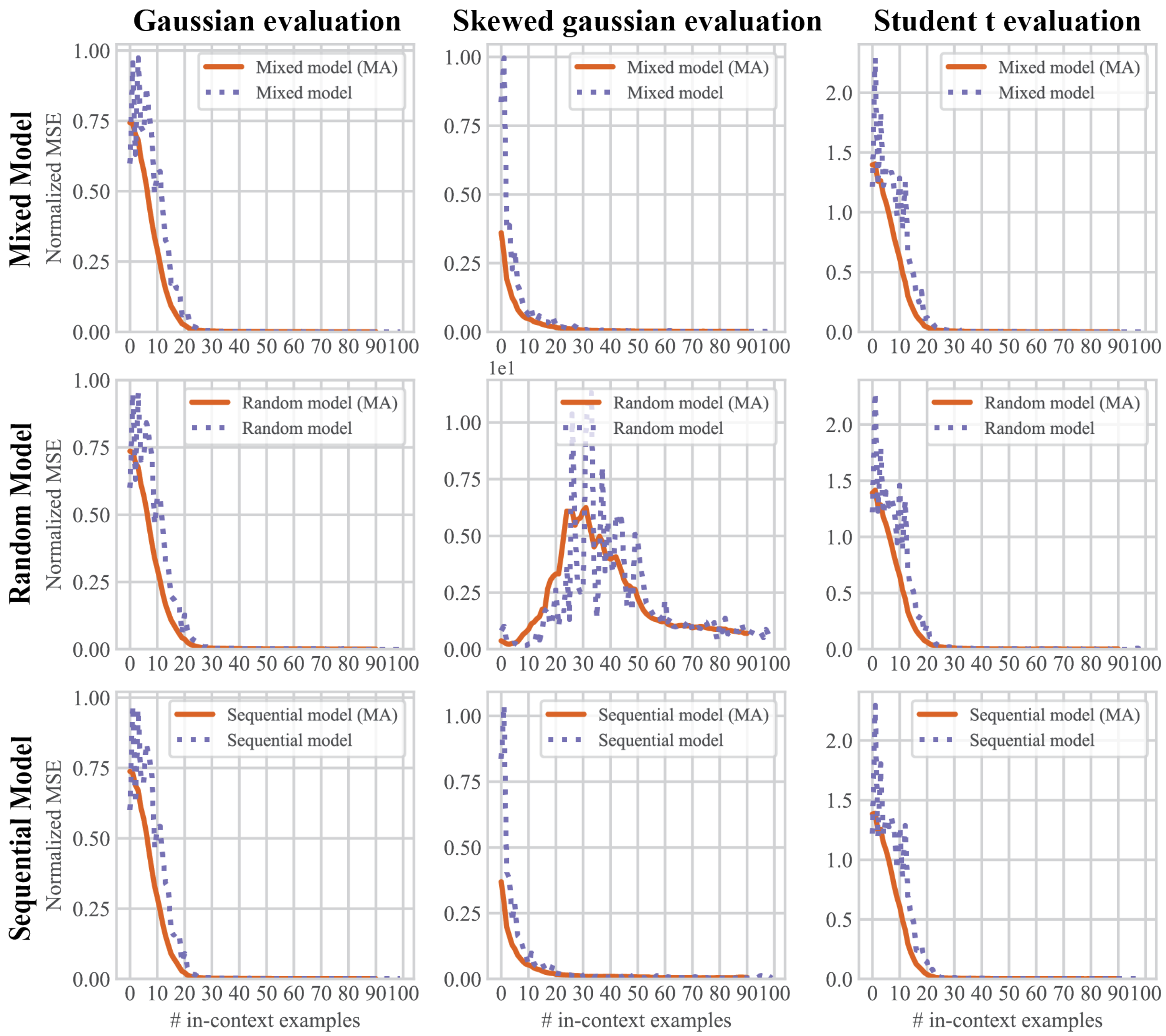

Figure 9: Normalized MSE over number of in-context examples for curriculum distribution learning models. Solid line represents the moving average (window = 10) whereas the dashed line is the actual value. Scientific notation is used for the y-axis.