Solving Empirical Bayes via Transformers

Anzo Teh, Mark Jabbour, Yury Polyanskiy*

February 17, 2025

Abstract

This work applies modern AI tools (transformers) to solving one of the oldest statistical problems: Poisson means under empirical Bayes (Poisson-EB) setting. In Poisson-EB a high-dimensional mean vector θ (with iid coordinates sampled from an unknown prior π) is estimated on the basis of $X = \text{Poisson}(\theta)$. A transformer model is pre-trained on a set of synthetically generated pairs (X, θ) and learns to do in-context learning (ICL) by adapting to unknown π. Theoretically, we show that a sufficiently wide transformer can achieve vanishing regret with respect to an oracle estimator who knows π as dimension grows to infinity. Practically, we discover that already very small models (100k parameters) are able to outperform the best classical algorithm (non-parametric maximum likelihood, or NPMLE) both in runtime and validation loss, which we compute on out-of-distribution synthetic data as well as real-world datasets (NHL hockey, MLB baseball, BookCorpusOpen). Finally, by using linear probes, we confirm that the transformer's EB estimator appears to internally work differently from either NPMLE or Robbins' estimators.

Contents

*M.J. was with the Department of EECS, MIT, Cambridge, MA, email: mjabbour@mit.edu. A.T. and Y.P. are with the Department of EECS, MIT, Cambridge, MA, email: anzoteh@mit.edu and yp@mit.edu.

1 Introduction

Transformers have received a lot of attention due to the prevalence of large language models (LLM). More generally, we think of (encoder-only) transformers as generic engines for learning from exchangeable data. Since most classical statistical tasks are formulated under iid sampling assumption, it is very natural to try to apply transformers to them [GTLV22].

Training transformers for classical statistical problems serves two purposes. One is obviously to get better estimators. Another, equally important, goal of such exercises is to elucidate the internal workings of transformers in a domain with a much easier and much better understood statistical structure than NLP. In this work, we believe, we found the simplest possible such statistical task: *empirical Bayes (EB) mean estimation.* We believe transformers are suitable for EB because EB estimators naturally exhibit a shrinkage effect (i.e. biasing mean estimates towards the nearest mode of the prior), and so do transformers, as shown in [GLPR24] that the attention mechanisms tend to cluster tokens. Additionally, the EB mean estimation problem is permutation equivariant, removing the need for positional encoding. In turn, estimators for this problem are in high demand [KG24, GK23, GK22] and unfortunately the best classical estimator (so-called non-parametric maximum likelihood, or NPMLE) suffers from slow convergence. In this work, we demonstrate that transformers outperform NPMLE while also running almost 100x faster. We now proceed to defining the EB task.

Poisson-EB task: One observes n samples $X_1, \dots, X_n$ which are generated iid via a two-step process. First, $\theta_1, \dots, \theta_n$ are sampled from some unknown prior π on $\mathbb{R}$. The π serves as an unseen (non-parametric) latent variable and we assume nothing about it (not even continuity or smoothness). Second, given θ_i's, we sample X_i's conditionally iid via $X_i \sim \mathrm{Poi}(\theta_i)$. The goal is to estimate $\theta_1, \cdots, \theta_n$ via $\hat{\theta}_1, \cdots, \hat{\theta}_n$ upon seeing $X_1, \cdots, X_n$ that minimizes the expected mean-squared error (MSE), $\mathbb{E}[(\hat{\theta}(X) - \theta)^2]$. If π were known, the Bayes estimator that minimizes the MSE is the posterior mean of θ, which also has the following form.

$$\hat{\theta}_\pi(x) = \mathbb{E}[\theta | X = x] = (x+1)\frac{f_\pi(x+1)}{f_\pi(x)} \,. \tag{1}$$

where $f_\pi(x) \triangleq \mathbb{E}_\pi[e^{-\theta}\frac{\theta^x}{x!}]$ is the posterior density of x. Given that π is unknown, an estimator π can only instead approximate $\hat{\theta}_\pi$. We quantify the quality of the estimation as the *regret*, defined as the excess MSE of $\hat{\theta}$, over $\hat{\theta}_\pi$.

$$\begin{aligned}\mathsf{Regret}(\hat{\theta}) &= \mathbb{E}\left[\left(\hat{\theta}(X) - \theta\right)^2\right] - \mathbb{E}\left[\left(\theta_\pi(X) - \theta\right)^2\right] \\ &= \mathbb{E}\left[\left(\hat{\theta}(X) - \theta\right)^2\right] - \mathsf{mmse}(\pi)\end{aligned}$$

In this Poisson-EB setting, multiple lines of work have produced estimators that resulted in regret that vanishes as sample size increases [BGR13, PW21, JPW22, JPTW23]. Robbins estimator [Rob51, Rob56] replaces the unknown posterior density f_π in (1) with $N_n(\cdot)$, the empirical count among the samples $X_1, \cdots, X_n$. Minimum distance estimators first estimate a prior (e.g. the NPMLE estimator $\hat{\pi}_{\mathsf{NPMLE}} = \text{argmax}_Q \prod_{i=1}^n f_Q(X_i)$), and then produces the plugged-in Bayes estimator $\hat{\theta}_{\hat{\pi}}$. Notice that Robbins estimator suffers from multiple shortcomings like numerical instability (c.f. [EH21]) and the lack of monotonicity property of Bayes estimator $\hat{\theta}_\pi$ (c.f. [HS83]), while minimum-distance estimators are too computationally expensive and do not scale to higher dimensions. [JPTW23] attempts to remedy the 'regularity vs efficiency' tradeoffs in these estimators with an estimator based on score-estimation equivalent in the Poisson model. However, despite the monotone regularity added, this estimator still does not have a Bayesian form: a cost one pays to achieve an efficient computational time.

Solving Poisson-EB via transformers. We formulate our procedure for solving Poisson-EB as follows: we generate synthetic data and train our transformers on those. Then, we freeze their weights and present new data to be estimated. To our knowledge, this is the first line of work that studies using neural network models for empirical Bayes. Concretely, our contributions are as follows:

1. In Section 4, we show that transformers can approximate Robbins and the NPMLE via the universal approximation theorem. We also use linear probes to show that our pre-trained transformers work differently than the two aforementioned estimators.

2. In Section 5, we set up synthetic experiments to demonstrate that synthetically pre-trained transformers can generalize to unseen sequence lengths and evaluation priors. This is akin to [XRLM21] where ICL occurs at test time despite distribution mismatch.

3. In Section 6, we evaluate these transformers on real datasets for a similar prediction task to demonstrate that they often outperform the classical baselines and crush them in terms of speed.

One significance of our synthetic experiments is that transformers demonstrate *length-generalization* by achieving lower regret upon being tested on sequence lengths up to 4x the length they are trained on, even on unseen priors. This comes as multiple works show mixed success of length-generalization of transformers [ZAC$^+$24, WJW$^+$24, KPNR$^+$24, AWA$^+$22].

We mention that there is a long literature studying transformers for many statistical problems [BCW$^+$24]. What makes this work different is that a) our estimator does not just match but improve upon existing (classical) estimators, thus advancing the statistical frontier; b) our setting is unsupervised and much closer to NLP compared to most previous work considering supervised learning (classification and regression), in which data comes in *pairs*, thus requiring unnatural tricks to pair tokens; c) our problem is non-parametric.

In summary, we demonstrate that even for classical statistical problems, transformers offer an excellent alternative (in runtime and performance). For the simple 1D Poisson-EB task, we also found that already very economically sized transformers (<100 k parameters) can have excellent performance.

2 Related Work

Transformers and in-context learning (ICL). Transformers have shown the ability to do ICL, as per the thread of work summarized in [DLD$^+$22]. ICL is primarily manifested in natural language processing [BMR$^+$20, DSD$^+$22] and learning linear models [ASA$^+$22, ZFB23]. Other examples that transformers can learn are gradient descent [BCW$^+$24], several non-linear function classes [GTLV22], and support vector machine [TLTO23], while having limited ability on boolean functions [BPBK23]. Recent works have also explained ICL from the Bayesian point of view [MHH24, PAG23], including showing Bayesian behavior even upon train-test distribution mismatch [XRLM21].

How do transformers work? [YBR$^+$19] have established the universal approximation theorem of transformers. This was later extended for sparse transformers [YCB$^+$20] and ICL setting [FdHP24]. Its limitations are further discussed in [NKB24]. Transformers have also been shown to do other approximation tasks, like Turing machines [WCM22, PBM21]. From another perspective, [AB18] introduces linear probes as a mechanism of understanding the internals of a neural network, which is further studied in [Bel22]. Linear probe has also been applied in transformers to study its ability to perform NLP tasks [TDP19], achieve second order convergence [FCJS24], and learn various functions in-context [GHM+23]. One such application is ICL linear regression to look for moments [ASA+22]. Recently, linear probe has been used by [AZR+24] to improve in-context learning.

Empirical Bayes. Empirical Bayes is a powerful tool for large-scale inference [Efr12]. Some of its applications include performing downstream tasks like linear regression [KWCS24, MSS23], estimating the number of missing species [FCW43], and large scale hypothesis testing [ETST01]. In computational biology, empirical Bayes has also been used in sequencing frameworks [HK10, LDT+13], though these frameworks are mostly parametric and rely on estimating the parameters of a prior.

In the theoretical setting, multiple lines of work have established the theoretical bounds that can be achieved by empirical Bayes estimators. In the Poisson-EB problem, Robbins [Rob51, Rob56] formulated an estimator based on Tweedie's formula, known as f-modelling. In the normal means EB problem, [JZ09] formulated a g-modelling approach via prior estimation, which was also adapted to the Poisson-EB problem. More recently, [JPTW23] formulated an estimator based on ERM on monotone functions, which introduces regularity to the estimators while also escaping the computationally expensive prior estimation process. The optimality of these estimators has been established in the following works: [BGR13, PW21, JPW22, JPTW23].

3 Preliminaries

3.1 Baselines description

We outline some of the classical algorithms that we will be benchmarking against.

Non empirical Bayes baselines. When nothing is known about the prior π the minimax optimal estimator is the familiar maximum-likelihood (MLE) estimator $\hat{\theta}_{\mathsf{MLE}}(x) = x$. However, when one restricts priors in some way, the minimax optimal estimator is not MLE, but rather a Bayes estimator for the *worst-case* prior. In this work, we consider priors restricted to support $[0, 50]$. The minimax optimal estimator for this case is referred to as the *gold standard* (GS) estimator to signify its role as the "best" in the sense of classical (pre-EB) statistics. Appendix A.1 contains derivation of GS.

Empirical Bayes baselines. We will use the following empirical Bayes estimators as introduced in Section 1: the Robbins estimator, NPMLE estimator, and the ERM-monotone estimator with algorithm described in Lemma 1 of [JPTW23].

3.2 Transformer Architecture

Next, we describe our transformer architecture, which closely mimics the standard transformer architecture in [VSP+17]. Given the permutation invariance of the Bayes estimator, we do not use positional encoding or masking. Thus effectively, it is a full-attention encoder-only transformer with one linear decoder on top.

One aspect worth mentioning is that at the encoding stage, we are using *two* different weights, split evenly across the N layers. The intuition behind it is that one learns the encoding part (input) and the other the decoding part (output).

3.3 Training Protocol

Data generation. We emphasize that all our transformers are trained on synthetic data, using the Poisson-generated integers X as inputs and the hidden parameters θ as labels. We use the plain vanilla MSE loss $\sum(\hat{\theta}(X_i) - \theta_i)^2$. There are two classes of priors from which we generate θ, the neural-generated prior-on-priors, and Dirichlet process with base distribution $\mathsf{Unif}[0, 50]$ within each batch. We fix the sentence length $= 512$ throughout training. With the exception as noted later in Section 5.1, we cap the label at $\theta_{\max} = 50$

(i.e. our priors are in the class $\mathcal{P}([0,50])$). We defer the detailed discussion to Appendix A.2, including the motivation to train with a mixture of the two priors.

Parameter Selection. We consider models of 6, 12, 18, 24, and 48 layers, embedding dimension dmodel either 32 or 64, and number of heads in 4, 8, 16, 32. We fix the number of training epochs to 50k, the learning rate to 0.02, and the decay rate every 300 epochs to 0.9. Among the trained models, we chose our models based on the mean-squared error evaluated on neural prior-on-prior and Dirichlet process during inference time. We then arrive at the two models described in Table 1, which we will name T18 and T24 depending on their number of layers. Both have around 25.6k parameters. We also define T18r and T24r as the transformers we train with random $\theta_{\max}$. [1]

Table 1: The characteristics of T18 and T24, respectively.

Transformer	Layers	Embedding dimension	# Heads	$\theta_{\max}$
T18	18	32	16	50
T24	24	32	8	50
T18r	18	32	16	Random $[10,150]$
T24r	24	32	8	Random $[10,150]$

4 Understanding transformers

In this section, we try to gain an intuition on how transformers work in our setting. We achieve this from two angles. First, we establish some theoretical results on the expressibility of transformers in solving empirical Bayes tasks. Second, we use linear probes to study the prediction mechanism of the transformers.

4.1 Expressibility of Transformers

We discuss the feasibility of using transformers to solve the empirical Bayes prediction task. Indeed, the study of the universal approximation theorem has been done on multilayer perceptron, c.f. [Aug24], with some variations like bounded weights [GI18] and width [KL20, PYLS20]. More recently, universal learnability of transformers has been established, first in [YBR$^+$19], which shows that 2 heads, each of size 1, and 4 hidden dimensions are all we need. [FdHP24] further characterizes universal learnability in terms of in-context learning.

To start with, we consider the clipped Robbins estimator, defined as follows:

$$\hat{\theta}_{\mathsf{Rob},d,M}(x) = \begin{cases} \min\{(x+1)\frac{N(x+1)}{N(x)}, M\} & x < d \\ M & x \geq d \end{cases} \tag{2}$$

Here, we show that transformers can learn this clipped Robbins estimator up to an arbitrary precision.

Theorem 4.1. *Set a positive integer d and a positive real number M. Then for any $\epsilon > 0$, there exists a transformer architecture with one encoding layer, skip connection, and embedding dimension $d+1$ that learns the clipped Robbins estimator $\hat{\theta}_{\mathsf{Rob},d,M}$ up to a precision ϵ.*

Similarly, we may show that transformers can approximate NPMLE up to an arbitrary input value and precision.

Theorem 4.2. *Let $M > 0$, and denote the NPMLE estimator $\hat{\theta}_{\mathsf{NPMLE},M}$, the NPMLE estimator chosen among $\mathcal{P}([0,M])$. For each integer $d > 0$ consider the following modified NPMLE function:*

$$\theta_{\mathsf{NPMLE},d,M}(x) = \begin{cases} \hat{\theta}_{\mathsf{NPMLE}}(x) & x \leq d \\ M & x > d \end{cases}$$

[1] In the future, we will add T18r and T24r to all comparisons, but for now they only appear on Fig. 5 and Section 5.1.

then for any $\epsilon > 0$ there exists a transformer network that can approximate $\theta_{\mathsf{NPMLE},d}$ uniformly up to ϵ-precision.

Full proofs are deferred to Appendix B and we only give a sketch for now. For Robbins approximation, we create an encoding mechanism that encodes $\frac{N(X_i)}{N(X_i)+(X_i+1)N(X_i+1)}$ at position i among $1, \cdots, n$ and use a decoder to approximate the function $x \to \min\{\frac{1}{x} - 1, M\}$. For NPMLE approximation, we pass in the Sigmoid of the integer inputs as embedding, and show that $\hat{\theta}_{\mathsf{NPMLE}}$ can be continually extended, with sigmoid-transformed empirical distribution as arguments. For the encoding part, we provide a pseudocode in Appendix C that closely follows PyTorch's implementation.

To illustrate the significance of both of these theorems, we demonstrate that transformers can learn an empirical Bayes prediction task to an arbitrarily low regret.

Corollary 4.3. *For any $\epsilon > 0$, there exists an integer N and a transformer network Γ such that for all $n \geq N$, the minimax regret of $\Gamma(X_1, \cdots, X_n)$ on prior $\pi \in \mathcal{P}([0, \theta_{\max}])$ satisfies*

$$\sup_{\pi \in \mathcal{P}([0,\theta_{\max}])} \mathsf{Regret}(\Gamma(X_1, \cdots, X_n)) \leq \epsilon$$

4.2 How do transformers learn?

We study the mechanisms by which transformers learn via linear probe [AB18]. To this end, we take the representation of each layer of our pretrained transformers, and train a decoder that comprises a layer normalization operation, linear layer, and GeLU activation. This decoder is then trained with the following labels: frequency $N(x)$ within a sequence, and posterior density $f_{\hat{\pi}}(x)$ estimated by the NPMLE. The aim is to study whether our transformers function like the Robbins or NPMLE. In the plot in Fig. 1, we answer this as negative, showing that our transformers are not merely learning about these features, but instead learning what the Bayes estimator $\hat{\theta}_\pi$ is.

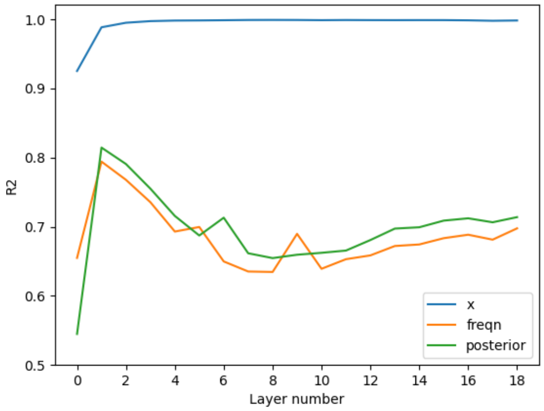


Figure 1: R^2 score of linear probe results against $N(x)$,$f_\pi(x)$ and x for T18 (the plots for T24 appear similar). We see that while x itself is easily recoverable from any layer, "knowledge" about the former two quantities appears to decrease with depth.

5 Synthetic Experiments

We now evaluate our trained transformers on the following: How well do they generalize? This can be done by evaluating on the following: sequence lengths other than the ones we have trained on, unseen priors, and unknown bound $\theta_{\max}$. We also compare against the classical algorithms introduced in Section 3 to demonstrate the superiority of these transformers by showing the average regret; most other details are deferred to Appendix D.1. We also investigate the inference time to show its advantage over NPMLE.

5.1 Ability to Generalize

Adaptibility to various sequence lengths. In this experiment, we evaluate the ability of transformers to adapt to different sequence lengths, both fewer than and more than what is trained. To do so, we evaluate them on 4096 neural prior-on-priors (which is part of the training distribution), but on various sequence length n: 128, 256, 512, 1024, and 2048. For each such prior, we generate 192 batches for evaluation. We report the average regret over the 4096 priors in Fig. 2.

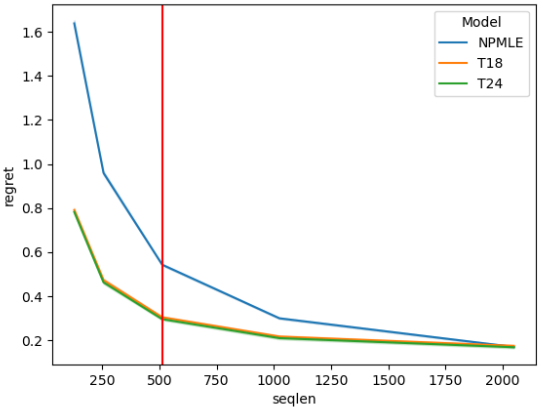


Figure 2: Regret vs sequence length (neural prior). The regret decreases for both transformers as the sequence length increases, showing that they do have the ability to generalize. We nevertheless note that NPMLE has a better generalization ability, as shown by the regret at sequence length 2048 as compared to smaller sequences. In comparison, the average regret for ERM monotone is 11.20, 8.19, 5.58, 3.66, and 2.36 for the various sequence lengths, while the average regret for MLE and GS stays constant at 14.816 and 14.658, respectively.

Robustness against unseen priors. Our transformers are trained on a mixture of neural and Dirichlet priors. Here, we consider their performances on the worst case prior in $\mathcal{P}([0, 50])$ as mentioned in Section 3.1 and further explained in Appendix A.1. The numbers of batches we use in this prior are 786k (for sequence lengths $n = 128, 256, 512$), 393k (for $n = 1024$), and 197k (for $n = 2048$). We also consider another unseen prior-on-prior: the multinomial prior supported on $[0, 50]$ with fixed, evenly split grids and weights distributed as Dirichlet distribution, using sequence lengths 512, 1024, and 2048, using 192 batches for each of the 4096 priors we evaluate on. We report the estimated regret in Fig. 3 and Fig. 4 to show that transformers produce regret comparable to the strongest alternative (NPMLE).

Training under randomized $\theta_{\max}$. In another experiment, we investigate the effect of mismatched $\theta_{\max}$. on the performance of transformers without knowledge of $\theta_{\max}$. Specifically, we train two sets of transformers, one as reported in Table 1, the other set (T18r, T24r) with the same parameters but with $\theta_{\max}$

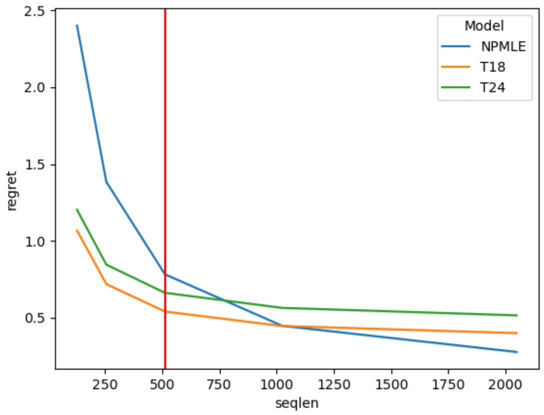


Figure 3: Regret of various transformers on worst prior compared to NPMLE. Again, transformers show the ability to generalize to longer sequence lengths, although for longer sequences NPMLE generalizes better. Note that this is already better than ERM-monotone's regret at 12.79, 9.80, 6.99, 4.81, and 3.26 across the 5 sequence lengths, while MLE's regret stays at 11.73.

randomized according to the following mixture:

$$\theta_{\max} \sim \frac{3}{4}\mathcal{N}(0, 50, 10^2) + \frac{1}{8}\text{Exp}(50) + \frac{1}{8}\text{Cauchy}(50, 10)$$

and clamped at $[10, 150]$. Then, for the two sets of transformers, we evaluate them on 4096 neural prior-on-priors, using the default sequence length = 512 and 192 batches for each prior. We report the distribution of regrets in Fig. 5 which demonstrates that transformers trained with randomized $\theta_{\max}$ see a small deterioration in regret, but nonetheless still outperform NPMLE in regret minimization.

5.2 Inference Time Comparison

We evaluate their inference time of various estimators over 4096 neural prior-on-priors, where for each prior we consider the time needed to estimate the hidden parameter of 192 batches and sequence length 128, 256, 512, 1024, and 2048. Each program is given 2 Nvidia Volta V100 GPUs and 40 CPUs for computation. The results are tabulated at Fig. 6, where we see that the transformers' runtime is comparable to that of ERM's.

6 Real Data Experiments

In this section, we answer the following question: Can our transformers that are pre-trained on synthetic data perform well on real datasets without re-training on any part of the real datasets?

To do so, we consider the following experimental setup: Given an integer-valued attribute, let X be the count of the attribute in the initial section we observe, and Y be the count of a similar attribute in the remaining section that we should predict. We assume that given a horizon length (duration, sentence length, etc) n_X and n_Y of the two sections, there exist hidden parameters θ_i such that $X_i \sim \text{Poi}(n_X\theta_i)$ and $Y_i \sim \text{Poi}(n_Y\theta_i)$, independently (for convenience we will scale θ_i such that $n_X = 1$). Our goal is to predict $\hat{Y} = n_Y\hat{\theta}(X)$ using empirical Bayes methods. We will focus on the following two types of datasets: sports and word frequency. Below, we describe the types of datasets that we would study. Throughout this section, we name (X, Y) as the input and label sets, respectively.

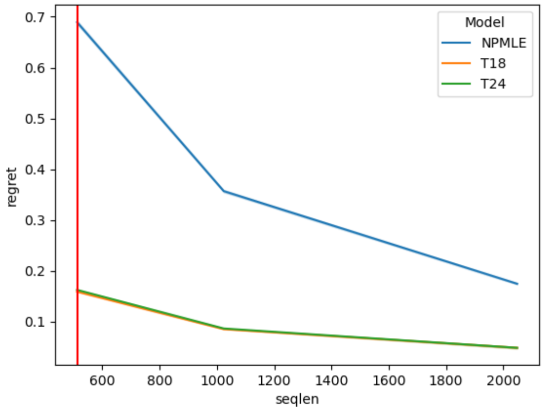


Figure 4: Average regret of various transformers on multinomial priors compared to NPMLE. Here, transformers show ≥ 2 times improvement over NPMLE even on sequence lengths it never trained on. In comparison, ERM-monotone's regret across sequence lengths are at 5.17, 3.69, 2.57, while MLE's and GS's regrets stay at 5.87 and 5.63, respectively.

6.1 Sports datasets

Here, X and Y are the numbers of goals scored by a player within disjoint and consecutive timeframes, and θ represents the innate ability of the given player. We will consider two datasets: National Hockey League (NHL) and Major League Baseball (MLB).

NHL dataset. We proceed in the same spirit as [JPW22], Section 5.2, and study the data on the total number of goals scored by each player in the National Hockey League for 29 years: from the 1989-1990 season to the 2018-2019 season (2004-2005 season was canceled). The data is obtained from [Hoc00], and we focus on the skaters' statistics. Here, given the number of goals a player scored in season j, we wish to predict the same for season $j+1$ (thus the input and label sets are the number of goals a player scored in consecutive seasons, and $n_Y = 1$). We study the prediction results when fitting all players at once, as well as fitting only positions of interest (defender, center, and winger).

MLB dataset. The dataset is publicly available at [Ret96], and can be processed by [Est18]. Here, we study the hitting count of each player in batting and pitching players from 1990 to 2017. Unlike the between-season prediction as we did for the NHL dataset, we do in-season prediction. That is, we take X as the number of goals scored by a player in the beginning portion of the season, and Y in the rest of the season. For batting and pitching players (which we fit separately), we use X as the goals in the first $\frac{1}{5}$ and first $\frac{1}{6}$ of the season (i.e. $n_Y = 4, 5$), respectively.

6.2 Word frequency datasets

In this setting, we model the alphabet of tokens as M categorical objects $A = \{A_1, \cdots, A_M\}$. Given n samples from these objects, and denote $(X_1, \cdots, X_M)$ the frequency of the samples. Suppose we are to estimate the frequencies $(Y_1, \cdots, Y_M)$ of an unseen section of length t (here t known). We model as follows: consider $p_1, \cdots, p_M$ as the "inherent" probability distribution over M (or proportion in a population), so $\sum_{i=1}^{M} p_i = 1$. Now the frequency $X_i \sim \text{Binom}(n, p_i)$, which we may instead approximate as $X_i \sim \text{Poi}(np_i)$. Thus we may use empirical Bayes method to estimate $\hat{\theta}_i = n\hat{p}_i$ based on the frequencies $X_1, \cdots, X_M$, and then predict $\hat{Y}_i = \frac{t}{n}\hat{\theta}_i$.

BookCorpusOpen. BookCorpus is a well-known large-scale text dataset, originally collected and ana-

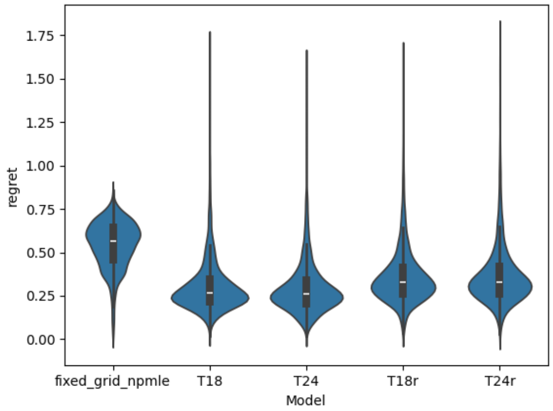


Figure 5: Comparison of regrets when $\theta_{\max}$ is trained randomly. The mean regret increases by 18.5% and 23.4% for the transformers with 18 and 24 layers, respectively, when compared against transformers that learn the true $\theta_{\max}$ during training, but still outperforms NPMLE. All comparisons resulted in a significant t-score ($p <$1e-100).

lyzed by [ZKZ$^+$15]. Here, we use a newer version named BookCorpusOpen, hosted on websites like [Dil24]. This version of the dataset contains 17868 books in English; we discard 6 of the books that are too short ($\leq$ 2000 tokens), and 5 other books where NPMLE incurs out-of-memory error. To curate the dataset, we first tokenize the text using scikit-learn's CountVectorizer with English stopwords removed. For each book, the input set comprises the beginning section containing approximately 2000 tokens, while the label set the remainder of the book. Then for each word, X and Y are the frequency of each word within the input and label set, respectively. We will then use the prediction $\hat{Y} = n_Y \cdot \hat{\theta}(X)$ where n_Y is the ratio of the number of sentences in the label set to that of the input set.

6.3 Evaluation Methods

We will use the RMSE of each dataset item, normalized by n_Y, as our main evaluation metric. Specifically, for each dataset, we compute the RMSE incurred by each estimator. We then compare them using the following guidelines.

Comparison against MLE. We consider the ratio of RMSE of each estimator against that of the MLE, and ask, "how much improvement did we achieve against the MLE" by looking at the *average* of the ratio.

Relative ranking. We use the Plackett-Luce [Pla75, Luc59] ranking system to determine how well one estimator ranks over the other.

Significance of improvement. We will also consider whether one improvement is *significant* by performing paired t-test on the RMSE of transformers against the baselines.

We tabulate our findings in Table 2 and Table 3. In addition, we also show a few violin plots in Fig. 7, Fig. 8, Fig. 9 for NHL, MLB batting, and Bookcorpus to supplement Table 2 (with Robbins removed due to its wide variance). From Table 3, we conclude a nontrivial improvement of the transformers over the classical methods in most of the datasets. A more detailed comparison (e.g. the ELO rating of estimators' RMSE and the MAE metric), is shown in Appendix D.2.

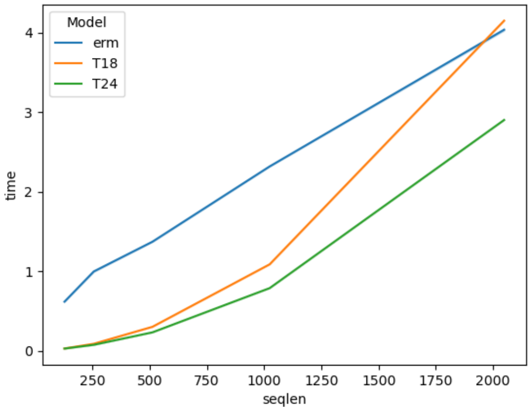


Figure 6: Time vs sequence length, showing that the inference time of transformers is comparable with that of ERM monotone. We nevertheless qualify this finding by noting that these running times seem to scale superlinearly with sequence length. For comparison, the running time of NPMLE for sequence lengths 128, 256, 512, 1024, and 2048 are 41.69, 67.70, 109.81, 175.72, and 289.79 seconds, respectively, which indicates that transformers are 2 orders of magnitudes faster than NPMLE.

Table 2: 95% confidence interval of the percentage improvement of RMSE by each algorithm over MLE.

Dataset	Robbins	ERM	NPMLE	T18	T24
NHL	-30.55 ± 6.55	1.46 ± 0.65	3.24 ± 0.92	**3.51 ± 1.01**	3.46 ± 1.00
NHL (defender)	-19.54 ± 6.35	3.19 ± 1.32	6.48 ± 1.63	7.25 ± 1.88	**7.41 ± 1.86**
NHL (center)	-49.89 ± 10.36	0.38 ± 0.82	3.44 ± 0.94	**4.12 ± 1.14**	4.06 ± 1.07
NHL (winger)	-42.63 ± 7.58	0.76 ± 0.69	3.06 ± 0.87	**3.39 ± 1.03**	3.38 ± 1.01
MLB (batting)	-32.80 ± 5.67	2.50 ± 0.36	4.30 ± 0.41	4.45 ± 0.37	**4.58 ± 0.39**
MLB (pitching)	-21.71 ± 2.45	2.51 ± 0.31	4.70 ± 0.41	4.89 ± 0.42	**4.95 ± 0.38**
BookCorpusOpen	-4.58 ± 0.43	9.38 ± 0.10	10.82 ± 0.11	10.38 ± 0.18	**11.43 ± 0.17**

7 Conclusion and Future Work

We have demonstrated the ability of transformers to learn EB-Poisson via in-context learning. This was done by evaluating pre-trained transformers on synthetic data of unseen distribution and sequence length, and compared against baselines like the NPMLE. In this process, we showed that transformers can achieve decreasing regret as the sequence length increases. On the real datasets, we showed that these pre-trained transformers can outperform classical baselines in most cases.

One future direction will be to extend our work to multi-dimensional input, as discussed in [JPTW23] (Section 1.3), [JPW22] (Section 6). We believe that the transformers would still be able to learn the 'context' of the inputs in multi-dimensional settings. On the other hand, the g-modelling methods like the NPMLE can take $n^{\Theta(d)}$ inference time, which makes it not scalable. In addition, given that the focus of this work is on Poisson-EB, one natural direction is to extend it to the normal-means model [JZ09]. On the theoretical front, the expressibility and limitations of the transformers can be further studied, including settings where the model dimension is bounded. Finally, given that the focus has been studying transformers trained and evaluated on priors with compact support ($[0, 50]$ in our case), we plan to study further the behavior of

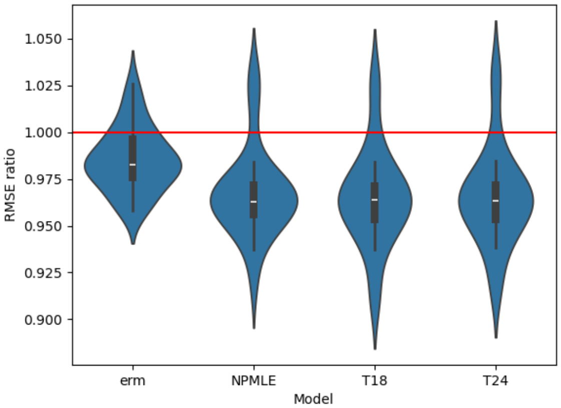


Figure 7: Violin plots of RMSE ratio achieved by multiple estimators over MLE on NHL.

Table 3: $\mathbb{P}[\text{RMSE(transformers)} > \text{RMSE(baselines)}]$ obtained via paired t-test.

	T18				T24			
Dataset	MLE	Robbins	ERM	NPMLE	MLE	Robbins	ERM	NPMLE
NHL	1.44E-06	2.51E-11	8.42E-06	0.120	1.57E-06	3.14E-11	8.96E-06	0.150
NHL (defender)	2.15E-08	9.76E-11	4.71E-08	8.72E-05	1.48E-08	9.67E-11	3.54E-08	4.77E-05
NHL (center)	1.14E-06	1.33E-11	6.01E-07	2.98E-03	6.38E-07	1.73E-11	5.61E-07	1.65E-03
NHL (winger)	1.76E-06	8.01E-13	1.33E-05	0.153	1.54E-06	8.29E-13	1.16E-05	0.152
MLB batting	1.39E-21	6.51E-14	1.30E-11	1.08E-03	8.49E-22	6.21E-14	1.93E-12	2.11E-08
MLB pitching	9.62E-20	1.21E-16	1.49E-12	2.34E-03	5.95E-21	1.13E-16	2.10E-13	4.78E-06
BookCorpusOpen	< 1E-100	< 1E-100	0.0104	1 - 1.19E-05	< 1E-100	< 1E-100	6.88E-15	0.133

transformers on priors with unbounded support (akin to how we did in one of the studies in Section 5.1).

Acknowledgements

This work was supported in part by the MIT-IBM Watson AI Lab and the National Science Foundation under Grant No CCF-2131115. Anzo Teh was supported by a fellowship from the Eric and Wendy Schmidt Center at the Broad Institute.

References

[AB18] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. 2018. *arXiv preprint arXiv:1610.01644*, 2018.

[ASA$^+$22] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? Investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

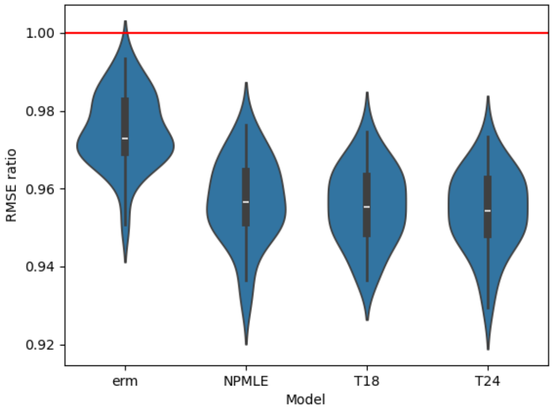


Figure 8: Violin plots of RMSE ratio achieved by multiple estimators over MLE on MLB batting.

[Aug24] Midhun T Augustine. A survey on universal approximation theorems. *arXiv preprint arXiv:2407.12895*, 2024.

[AWA+22] 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. *Advances in Neural Information Processing Systems*, 35:38546–38556, 2022.

[AZR+24] Momin Abbas, Yi Zhou, Parikshit Ram, Nathalie Baracaldo, Horst Samulowitz, Theodoros Salonidis, and Tianyi Chen. Enhancing in-context learning via linear probe calibration. In *International Conference on Artificial Intelligence and Statistics*, pages 307–315. PMLR, 2024.

[BCW+24] Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems*, 36, 2024.

[Bel22] Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[BGR13] Lawrence D Brown, Eitan Greenshtein, and Ya'acov Ritov. The Poisson compound decision problem revisited. *Journal of the American Statistical Association*, 108(502):741–749, 2013.

[BMR+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[BPBK23] Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and llms by learning to learn discrete functions. *arXiv preprint arXiv:2310.03016*, 2023.

[Dil24] Luca Diliello. BookCorpusOpen Dataset on Hugging Face, 2024. Accessed: 2024-11-09.

[DLD+22] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

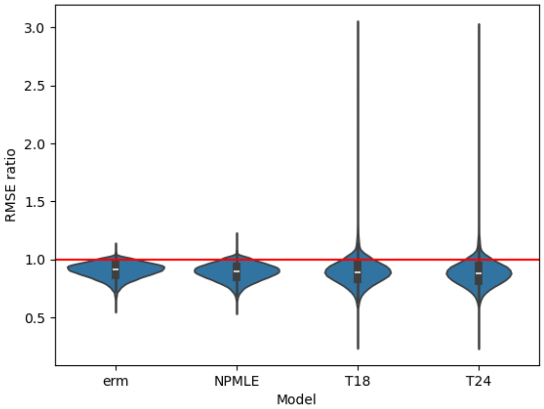


Figure 9: Violin plots of RMSE ratio achieved by multiple estimators over MLE on BookCorpusOpen.

[DSD+22] Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. Why can GPT learn in-context? Language models implicitly perform gradient descent as meta-optimizers. *arXiv preprint arXiv:2212.10559*, 2022.

[Efr12] Bradley Efron. *Large-scale inference: empirical Bayes methods for estimation, testing, and prediction*, volume 1. Cambridge University Press, 2012.

[EH21] Bradley Efron and Trevor Hastie. *Computer age statistical inference, student edition: algorithms, evidence, and data science*, volume 6. Cambridge University Press, 2021.

[Est18] Cal Estini. Retrosheet Repository on GitHub, 2018. Accessed: 2024-10-25.

[ETST01] Bradley Efron, Robert Tibshirani, John D Storey, and Virginia Tusher. Empirical bayes analysis of a microarray experiment. *Journal of the American statistical association*, 96(456):1151–1160, 2001.

[FCJS24] Deqing Fu, Tian-Qi Chen, Robin Jia, and Vatsal Sharan. Transformers learn to achieve second-order convergence rates for in-context linear regression. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[FCW43] Ronald A Fisher, A Steven Corbet, and Carrington B Williams. The relation between the number of species and the number of individuals in a random sample of an animal population. *The Journal of Animal Ecology*, pages 42–58, 1943.

[FdHP24] Takashi Furuya, Maarten V. de Hoop, and Gabriel Peyré. Transformers are Universal In-context Learners. *arXiv preprint arXiv:2408.01367*, 2024.

[GHM+23] Tianyu Guo, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. How do transformers learn in-context beyond simple functions? A case study on learning with representations. *arXiv preprint arXiv:2310.10616*, 2023.

[GI18] Namig J Guliyev and Vugar E Ismailov. On the approximation by single hidden layer feedforward neural networks with fixed weights. *Neural Networks*, 98:296–304, 2018.

[GK22] Jiaying Gu and Roger Koenker. Nonparametric maximum likelihood methods for binary response models with random coefficients. *Journal of the American Statistical Association*, 117(538):732–751, 2022.

[GK23] Jiaying Gu and Roger Koenker. Invidious comparisons: Ranking and selection as compound decisions. *Econometrica*, 91(1):1–41, 2023.

[GLPR24] Borjan Geshkovski, Cyril Letrouit, Yury Polyanskiy, and Philippe Rigollet. The emergence of clusters in self-attention dynamics. *Advances in Neural Information Processing Systems*, 36, 2024.

[GTLV22] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? A case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

[HK10] Thomas J Hardcastle and Krystyna A Kelly. baySeq: Empirical Bayesian methods for identifying differential expression in sequence count data. *BMC bioinformatics*, 11:1–14, 2010.

[Hoc00] Hockey-Reference. Hockey-Reference.com, 2000. Accessed: 2024-09-30.

[HS83] JC van Houwelingen and Th Stijnen. Monotone empirical bayes estimators for the continuous one-parameter exponential family. *Statistica Neerlandica*, 37(1):29–43, 1983.

[JPTW23] Soham Jana, Yury Polyanskiy, Anzo Z Teh, and Yihong Wu. Empirical Bayes via ERM and rademacher complexities: the poisson model. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 5199–5235. PMLR, 2023.

[JPW22] Soham Jana, Yury Polyanskiy, and Yihong Wu. Optimal empirical Bayes estimation for the Poisson model via minimum-distance methods. *arXiv preprint arXiv:2209.01328*, 2022.

[JZ09] Wenhua Jiang and Cun-Hui Zhang. General maximum likelihood empirical bayes estimation of normal means. *The Annals of Statistics*, 37(4):1647–1684, 2009.

[KG24] Roger Koenker and Jiaying Gu. Empirical bayes for the reluctant frequentist. *arXiv preprint arXiv:2404.03422*, 2024.

[KL20] Patrick Kidger and Terry Lyons. Universal Approximation with Deep Narrow Networks. In *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pages 2306–2327. PMLR, 09–12 Jul 2020.

[KPNR$^+$24] Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

[KWCS24] Youngseok Kim, Wei Wang, Peter Carbonetto, and Matthew Stephens. A flexible empirical Bayes approach to multiple linear regression and connections with penalized regression. *Journal of Machine Learning Research*, 25(185):1–59, 2024.

[LDT$^+$13] Ning Leng, John A Dawson, James A Thomson, Victor Ruotti, Anna I Rissman, Bart MG Smits, Jill D Haag, Michael N Gould, Ron M Stewart, and Christina Kendziorski. Ebseq: An empirical bayes hierarchical model for inference in rna-seq experiments. *Bioinformatics*, 29(8):1035–1043, 2013.

[Luc59] R Duncan Luce. *Individual choice behavior*, volume 4. Wiley New York, 1959.

[MHH24] Samuel Müller, Noah Hollmann, and Frank Hutter. Bayes' power for explaining in-context learning generalizations. *arXiv preprint arXiv:2410.01565*, 2024.

[MSS23] Sumit Mukherjee, Bodhisattva Sen, and Subhabrata Sen. A mean field approach to empirical Bayes estimation in high-dimensional linear regression. *arXiv preprint arXiv:2309.16843*, 2023.

[NKB24] Swaroop Nath, Harshad Khadilkar, and Pushpak Bhattacharyya. Transformers are expressive, but are they expressive enough for regression? *arXiv preprint arXiv:2402.15478*, 2024.

[PAG23] Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. *arXiv preprint arXiv:2306.04891*, 2023.

[PBM21] Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is turing-complete. *Journal of Machine Learning Research*, 22(75):1–35, 2021.

[Pla75] Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

[PW21] Yury Polyanskiy and Yihong Wu. Sharp regret bounds for empirical bayes and compound decision problems. *arXiv preprint arXiv:2109.03943*, 2021.

[PYLS20] Sejun Park, Chulhee Yun, Jaeho Lee, and Jinwoo Shin. Minimum width for universal approximation. *arXiv preprint arXiv:2006.08859*, 2020.

[Ret96] Retrosheet. Retrosheet Game Logs, 1996. Accessed: 2024-10-25.

[Rob51] Herbert Robbins. Asymptotically subminimax solutions of compound statistical decision problems. In *Proceedings of the second Berkeley symposium on mathematical statistics and probability*, pages 131–149. University of California Press, 1951.

[Rob56] Herbert Robbins. An Empirical Bayes Approach to Statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1956.

[TDP19] Ian Tenney, Das Dipanjan, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

[TLTO23] Davoud Ataee Tarzanagh, Yingcong Li, Christos Thrampoulidis, and Samet Oymak. Transformers as support vector machines. *arXiv preprint arXiv:2308.16898*, 2023.

[VSP$^+$17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[WCM22] Colin Wei, Yining Chen, and Tengyu Ma. Statistically meaningful approximation: A case study on approximating turing machines with transformers. *Advances in Neural Information Processing Systems*, 35:12071–12083, 2022.

[WJW$^+$24] Jie Wang, Tao Ji, Yuanbin Wu, Hang Yan, Tao Gui, Qi Zhang, Xuanjing Huang, and Xiaoling Wang. Length generalization of causal transformers without position encoding. *arXiv preprint arXiv:2404.12224*, 2024.

[XRLM21] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

[YBR$^+$19] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

[YCB+20] Chulhee Yun, Yin-Wen Chang, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. O(n) connections are expressive enough: Universal approximability of sparse transformers. *Advances in Neural Information Processing Systems*, 33:13783–13794, 2020.

[ZAC+24] Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. *arXiv preprint arXiv:2402.09371*, 2024.

[ZFB23] Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*, 2023.

[ZKZ+15] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. *arXiv preprint arXiv:1506.06724*, 2015.

A Detailed Discussion on Setups

A.1 Worst-case prior and Gold-Standard Estimator

We first define the worst-case prior and the gold-standard estimator.

Definition A.1 (Worst-case prior)**.** Let A be a compact subset of $\mathcal{R}$. Then the worst-case prior $\pi_{!,A}$ is defined as

$$\pi_{!,A} = \operatorname*{argmax}_{\pi \in \mathcal{A}} \mathsf{mmse}(\pi)$$

A sample distribution of the worst-case prior on $[0, 50]$ is illustrated in Figure 1 of [JPW22].

One motivation for using the worst case prior is that the Bayes estimator is considered the "gold standard" which minimizes the maximum-possible MSE across all priors supported on A. A concrete statement can be found in the following lemma.

Lemma A.2. *Let $\hat{\theta}_\pi$ be the Bayes estimator to a prior π. and let A be any compact subset of the reals. Then the least favorable prior $\pi_{!,A}$ of A satisfies the following:*

$$\mathsf{MSE}_{\delta_\theta}(\hat{\theta}_{\pi_{!,A}}) \leq \mathsf{mmse}(\pi_{!,A}), \forall \theta \in A$$

and equality holds whenever $\theta \in \mathsf{Supp}(\pi_{!,A})$.

This leads to the following corollary.

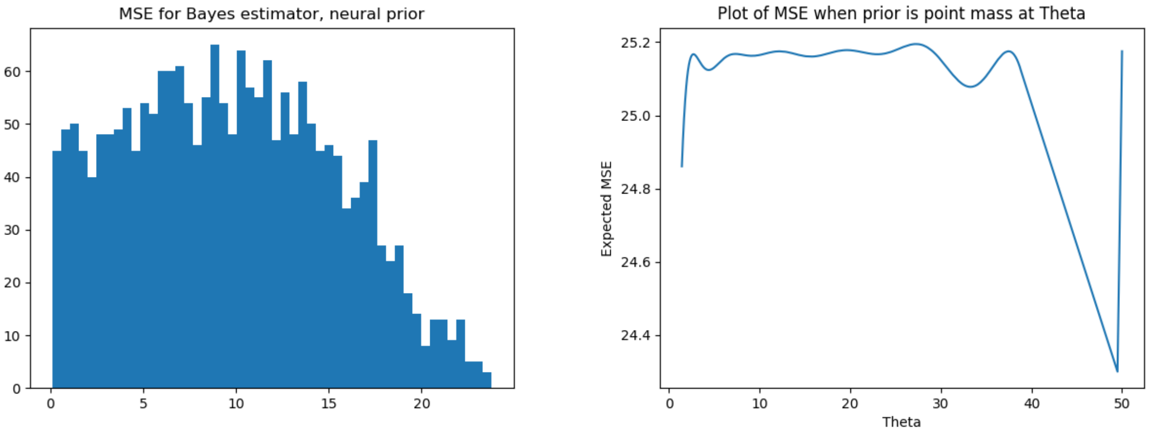
Corollary A.3. *For any compact subset A of the reals, we have*

$$\min_{\hat{\theta}} \max_{\pi \in \mathcal{P}(A)} \mathbb{E}[(\hat{\theta}(X) - \theta)^2] = \mathsf{mmse}(\pi_{!,A})$$

achieved by the Bayes estimator $f_{\pi_{!,A}}$ of the least favourable prior, $\pi_{!,A}$.

Proof. From Lemma A.2, we have $\mathsf{MSE}_\pi(\hat{\theta}_{\pi_{!,A}}) \leq \mathsf{mmse}(\pi_!)$ for any $\pi \in \mathcal{P}(A)$. Therefore $\min_{\hat{\theta}} \max_{\pi \in \mathcal{P}(A)} \mathbb{E}[(\hat{\theta}(X) - \theta)^2] \leq \mathsf{mmse}(\pi_!)$ by taking $\hat{\theta} = \hat{\theta}_{\pi_{!,A}}$. Now, for any $\hat{\theta}$, we have $\mathbb{E}_{\pi_{!,A}}[(\hat{\theta}(X) - \theta)^2] \geq \mathsf{mmse}(\pi_!)$. Therefore the conclusion follows. □

On the flip side, however, this estimator $f_{\pi!,A}$ does not leverage the fact the low-MMSE nature of some prior, leading to suboptimal regret produced by $f_{\pi!,A}$. Indeed, we consider the priors generated by the neural prior on prior protocols, and the histogram of MMSEs as shown in Fig. 10a. The MSE given by $f_{\pi!,[0,50]}$ on priors that are point masses as per Fig. 10b suggests that $f_{\pi!,[0,50]}$ is incapable of achieving low regrets on priors with low MMSEs.



(a) MMSE of neural priors in $\mathcal{P}([0,50])$

(b) MSE of $f_{\pi!,[0,50]}$ at point masses

Figure 10: Discussion on Worst Prior

A.2 Training Priors

We now offer a more detailed description of the training priors.

Neural-generated: prior on priors. We sample the hidden mean parameter θ via the following: first, let $\mathcal{M}$ be classes of priors determined by some two-layer perceptron with a non-linear activation in-between. This is concretely defined as:

$$\mathcal{M} = \{\pi : \pi = \varphi_{\sharp}^{W_1,W_2,\sigma}\mathsf{Unif}[0,1]\}$$

where $\varphi^{W_1,W_2,\sigma}(x) = \mathsf{Sigmoid}(10W_2\sigma(W_1x))$, W_1, W_2 are linear operators, and σ is an activation function chosen randomly from

$$GELU, ReLU, SELU, CELU, SiLU, Tanh, TanhShrink.$$

The parameter θ is then produced by sampling from a mixture of 4 priors in $\mathcal{M}$, and multiplied by $\theta_{\max}$ (or in the random $\theta_{\max}$ experiment as described in Section 5.1, each θ is then scaled differently).

Dirichlet process. Let the base distribution be defined as $H_0 \triangleq \text{Unif}([0,h])$. Within each batch the elements $\theta_1, \cdots, \theta_s$ are generated as follows:

$$\theta_j = \begin{cases} \theta_i & \text{w.p. } \frac{j-1}{\alpha+j-1}, \forall i = 1, \cdots, j-1 \\ x \sim H_0 & \text{w.p. } \frac{\alpha}{\alpha+j-1} \end{cases}$$

where α is a parameter that denotes how ‘close’ we are to iid generation ($\alpha = \infty$ essentially means we have iid). We use $\alpha = 50$ for a sequence length of 512. Note that Dirichlet process implies that our data is not generated iid for each batch, so the Bayes estimator has to be estimated differently. We omit the calculation of this Bayes estimator.

A.3 Why do we train using a mixture of two prior classes?

We consider the hypothesis: that our transformer trained under the mixture of the two priors is robust when evaluated under each of the priors. This can be verified via the following two tests: when evaluated on neural prior, is the performance (in terms of MSE) of the mixture-trained transformers closer to that of neural-trained ones as compared to the Dirichlet-trained ones? Similarly, when evaluated on Dirichlet prior, is the performance (in terms of MSE) of the mixture-trained transformers closer to that of Dirichlet-trained ones as compared to the neural-trained ones? Through the table of T-stat comparison done on the MSEs of 4096 seeds, we answer both these questions in the positive (the difference is especially obvious when evaluated on neural prior).

Table 4: Table of regret difference; $A - B$ denotes the difference of regret of transformers trained on A vs trained on B

	Evaluated on Neural		Evaluated on Dirichlet	
# lyr	mix − neu	dir − mix	mix − dir	neu − mix
12	0.0038	0.8645	0.0184	0.0379
18	0.0133	1.0647	0.0173	0.0469
24	0.0082	1.0021	0.0202	0.0388

B Technical Proofs

B.1 Approximation of known empirical Bayes baselines

Proof of Theorem 4.1. **Encoding step.** We embed our inputs representation $X \in \mathbb{R}^n$ into one-hot vector $Y \in \mathbb{R}^{n\times(d+1)}$ such that $Y_i = e_{X_i+1}$ if $X_i = 0, 1, \cdots, d$, and 0 otherwise. Then given sample size n, $Y \in \mathbb{R}^{(d+1)\times n}$. Now recall the following attention layer definition in (1) of [VSP+17]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where $Q = YW_Q, K = YW_K, V = YW_V$, and $W_Q, W_K \in \mathbb{R}^{(d+1)\times d_k}$. Let $Z = \text{Attention}(Q, K, V)$. We now design en encoding mechanism such that the representation after skip connection has the following:

$$(Y+Z)_{ij} = \begin{cases} 1 + \frac{N(X_i)}{N(X_i)+(X_i+1)N(X_i+1)} & j = X_i + 1 \le d \\ \frac{(X_i+1)N(X_i+1)}{N(X_i)+(X_i+1)N(X_i+1)} & j = X_i + 2 \le d \\ 1 & j = X_i + 1 = d \\ 0 & \text{otherwise.} \end{cases}$$

Define D be a large number, $W_Q = I_{d+1}$ the d-dimensional identity matrix, $W_V = \begin{pmatrix} I_d & 0 \\ 0 & 0 \end{pmatrix}$ and $W_K \in \mathbb{R}^{d\times d}$ satisfying

$$(W_K)_{i,j} = \begin{cases} D & i = j \\ D + \sqrt{d+1}\log i & j = i + 1 \\ 0 & \text{otherwise} \end{cases}$$

(thus $d_k = d + 1$). Then

$$(QK^T)_{i,j} = \begin{cases} D & X_i = X_j = d \\ D + \sqrt{k}\log(X_i + 1) & X_j = X_i + 1 \le d \; . \\ 0 & \text{otherwise} \end{cases}$$

Thus we have the following structure for $M \triangleq \text{Softmax}(S)$: $\text{row}_i(M) = \frac{1}{n}$ if $X_i \ge d + 1$, otherwise

$$M_{ij} = \begin{cases} \frac{1}{N(X_i)+(X_i+1)N(X_i+1)} & X_i = X_j \le d - 1 \\ \frac{X_i+1}{N(X_i)+(X_i+1)N(X_i+1)} & X_j = X_i + 1 \le d - 1 \\ 0 & \text{otherwise.} \end{cases}$$

Now given that $V = \begin{pmatrix}\mathrm{Col}_1(Y) & \cdots & \mathrm{Col}_d(Y) & 0\end{pmatrix}$, $Z_{ij} = \sum_{k:j=X_k+1} Z_{ik}$ for all $k \leq d-1$ (and 0 for k) This means:

$$Z_{ij} = \begin{cases} \frac{N(X_i)}{N(X_i)+(X_i+1)N(X_i+1)} & j = X_i + 1 \leq d-1 \\ \frac{(X_i+1)N(X_i+1)}{N(X_i)+(X_i+1)N(X_i+1)} & j = X_i + 2 \leq d \\ 0 & \text{otherwise.} \end{cases}$$

Thus adding back Y gives the desired output.

Decoding step. We define $Y_1 = \mathsf{ReLU}(Y + Z - 1)$, i.e. a linear operation (with bias) followed by the ReLU nonlinear operator. Notice that Z has entries all in $[0, 1]$, so Y_1 acts like $Y * Z$ (i.e. Z masked with Y). Let $Z' \in \mathbb{R}^n$ to be the row-wise sum of Y_1, i.e. $Z'_i = \frac{N(X_i)}{N(X_i)+(X_i+1)N(X_i+1)}$ if $X_i \leq d-1$ and 0 otherwise. Then we consider the following decoding function $f : [0, 1] \to [0, \theta_{\max}]$ by:

$$f(x) = \begin{cases} \frac{1}{x} - 1 & x \geq \frac{1}{1+M} \\ M & \text{otherwise} \end{cases}.$$

Then f is continuous, and $f(Z')$ is indeed $\hat{\theta}_{\mathsf{Rob},d,M}$. Therefore by universal approximation theorem, there exists a multilayer perceptron that approximates f within $[0, 1]$, as desired. □

Before proving Theorem 4.2, we need to establish the continuity of the clipped NPMLE, with arguments the sigmoid of the input integer and empirical distribution.

Lemma B.1. *Let $\varphi : \mathbb{R}_{\geq 0} \to [0, 1]$ be a strictly increasing and continuous function, and $Sig = \{\varphi(z) : z \in \mathbb{Z}\}$. Let $S = \sup(Sig)$ and $Sig^+ = Sig \cup \{S\}$. Denote $\tilde{\theta} : (\mathcal{P}(Sig^+) \times Sig^+ \to [0, M]$ be such that for each $p^{\mathsf{emp}} \in \mathcal{P}(\mathbb{Z}_{\geq 0})$ and $x \in \mathbb{Z}_{\geq 0}$, the function $\tilde{\theta}(\varphi_\sharp(p^{\mathsf{emp}}), \varphi(x)) = \hat{\theta}_{\mathsf{NPMLE},d,M}(p^{\mathsf{emp}}, x)$. Then $\tilde{\theta}$ can be extended into a function that is continuous in both arguments. (Here p^{emp} acts like an empirical distribution).*

Proof of Theorem 4.2. Starting with the input tokens $(X_1, \cdots, X_n)$, we consider the token-wise embedding $Y_i = \mathsf{Sigmoid}(X_i)$. Note that $\mathsf{Sigmoid}$ satisfies the assumption of φ in Lemma B.1. Denote p_n^{emp} as the empirical distribution determined by $(X_1, \cdots, X_n)$. By Lemma B.1, the function $\hat{\theta}_{\mathsf{NPMLE},d}(p_n^{\mathsf{emp}}, \cdot)$ can be continually extended (in the (weak*, ℓ_2) metric). Then [FdHP24], Theorem 1 says that there exists a transformers network Γ that satisfies

$$|\hat{\theta}(x_1, \cdots, x_n)_i - \Gamma(x_1, \cdots, x_n)_i| \leq \epsilon\,,$$

as desired. □

Proof of Lemma B.1. To establish continuity, it suffices to show that given a sequence of distributions $p_1^{\mathsf{emp}}, p_2^{\mathsf{emp}}, \cdots$ and integers $x_1, x_2, \cdots$ such that $\varphi_\sharp(p_n^{\mathsf{emp}}) \to \phi_0$ and $\varphi(x_n) \to y_0$ in (weak*, ℓ_2) metric, we have $\hat{\theta}_{\mathsf{NPMLE}}(p_n^{\mathsf{emp}}, x_n) \to \tilde{\theta}(\phi_0, y_0)$. Note that x_n are nonnegative integers, so given that φ is increasing and injective, either x_n is eventually constant (in which case $x_n \to x_0$ for some x_0), or $x_n \to \infty$ (in which case $y_0 = S \triangleq \sup(\mathrm{Sig})$). Note first that in the case $y_0 = S$ we have $\tilde{\theta}(\varphi_\sharp(p_n^{\mathsf{emp}}), \varphi(x_n)) = M$ for all n sufficiently large. Note also that p^{emp} is a distribution on nonnegative integers so $\varphi_\sharp(p_n^{\mathsf{emp}})(S) = 0$, which then follows that $\phi_0(S) = 0$ too. Thus $\phi_0 \in \mathcal{P}(\mathrm{Sig})$ and so there exists p_0 such that $\phi_0 = \varphi_\sharp(p_0)$.

It now remains to consider the case where $x_n = x_0$ for all sufficiently large n; w.l.o.g. we may even assume $x_n = x_0$ for all n. If $x_0 > d$ we are done since $\hat{\theta}_{\mathsf{NPMLE}}(p^{\mathsf{emp}}, x_0) = M$ for all π. Assume now that $x_0 \leq d$. Recall that $\hat{\theta}_{\mathsf{NPMLE}}(p^{\mathsf{emp}}, x_0) = (x_0 + 1)\frac{f_{\hat{\pi}}(x_0+1)}{f_{\hat{\pi}}(x_0)}$ where $\hat{\pi}$ is the prior estimated by NPMLE. Thus denoting $\hat{\pi}_n$ as NPMLE prior of p_n^{emp}, for each x it suffices to show that convergence of $f_{\hat{\pi}_n}(x)$. Now note that NPMLE also has the following equivalent form: $\hat{\pi}_n = \mathrm{argmin}_Q\, \mathrm{KL}(p_n^{\mathsf{emp}} || f_Q$, and note that KL can be written in the following form (c.f. Assumption 1 of [JPW22]).

$$\mathrm{KL}(\pi_1||\pi_2) = t(\pi_1) + \sum_{x \geq 0} \ell(\pi_1, \pi_2) \tag{3}$$

Notice that $\ell(a,b) := a\log\frac{1}{b}$ fulfills $b \to \ell(a,b)$ is strictly decreasing and strictly convex for $a > 0$. Fix $x_0 \le d+1$, we now have two subcases:

Case $p_0(x_0) > 0$**.** The claim immediately follows from that $\ell(p_0(x_0), b)$ is *strictly* convex in b, and that for each x we have $p_n^{\mathsf{emp}}(x) \to p_0(x)$ following the weak convergence of p_n^{emp}.

Case $p_0(x_0) = 0$**.** Let $Q_0 \in \operatorname{argmin}_Q t(p_0) + \sum_{x\ge 0} \ell(p_0(x), f_Q(x))$. By Theorem 1 of [JPW22], this Q_0 is unique. Now suppose that there is a subsequence $n_1, n_2, \cdots$ and a real number $\epsilon > 0$ such that

$$|f_{\hat{\pi}_{n_i}}(x_0) - f_{Q_0}(x_0)| > \epsilon$$

By the previous subcase, we have $f_{\hat{\pi}_{n_i}}(x) \to f_{Q_0}(x)$ for all $x \in \mathrm{Supp}(p_0)$. We now consider Q_1 as the solution to (3), but among the class of functions satisfying the constraint $|f_{Q_1}(x) - f_{Q_0}(x)| > \epsilon$. Such a constrained space is closed by proof of Theorem 1 in [JPW22], so there exists $\delta > 0$ such that

$$\begin{aligned}&\mathrm{KL}(p_0||f_{\hat{\pi}_{n_i}}) - \mathrm{KL}(p_0||f_{Q_0})\\ &\ge \mathrm{KL}(p_0||f_{Q_1}) - \mathrm{KL}(p_0||f_{Q_0}) \ge \delta\end{aligned}$$

On the other hand, by fixing Q, Q', we have

$$\begin{aligned}&(\mathrm{KL}(p_n^{\mathsf{emp}}||f_Q) - \mathrm{KL}(p_n^{\mathsf{emp}}||f_{Q'}))\\ &-(\mathrm{KL}(p_0||f_Q) - \mathrm{KL}(p_0||f_{Q'})) \to 0\end{aligned}$$

given that $p_n^{\mathrm{emp}} \to p_0$ weakly and that $\mathrm{KL}(p||f_Q) - \mathrm{KL}(p||f_{Q'}) = \sum_y p(y)\log\frac{f'_Q}{f_Q}$, which is a contradiction. □

Proof of Corollary 4.3. Choose d such that $\mathbb{P}[X > d] < \frac{\epsilon}{6\cdot\theta_{\max}^2}$. Note that there exists an N such that for $n \ge N$, both the Robbins estimator [BGR13] and NPMLE [JPW22], Theorem 3 enjoy a minimax regret bounded by $\frac{\epsilon}{6}$ over the class $\mathcal{P}([0,\theta_{\max}])$. Now, by the previous two theorems, there exists a transformers model Γ that can approximate either Robbins or NPMLE up to $\sqrt{\frac{\epsilon}{6}}$ precision uniformly for inputs up to d. Then we have

$$\begin{aligned}\mathrm{Regret}(\Gamma) &\le 2(\mathrm{Regret}(\hat{\theta}) + \mathbb{E}[(\hat{\theta}-\Gamma)^2])\\ &\le 2(\frac{\epsilon}{6} + \mathbb{E}[(\hat{\theta}(X) - \Gamma(X))^2\mathbf{1}_{\{X\le d\}}]\\ &+ \mathbb{E}[\theta_{\max}^2\mathbf{1}_{\{X>d\}}])\\ &\le 2(\frac{\epsilon}{6} + \frac{\epsilon}{6} + \frac{\epsilon}{6})\\ &= \epsilon\end{aligned}$$

□

B.2 Identities on Worst Prior

Proof of Lemma A.2. We consider the prior $\pi_\epsilon \triangleq (1-\epsilon)\pi_! + \epsilon\delta_{\theta_0}$ for some $\theta_0 \in A$. Then $\frac{\partial}{\partial\epsilon}\mathsf{mmse}(\pi_\epsilon)|_{\epsilon=0} \le 0$ with equality if $\theta_0 \in \mathrm{supp}(\pi_{!,A})$. Consider, now, the following form:

$$\begin{aligned}\mathsf{mmse}(\pi) &= \mathbb{E}[\theta^2] - \mathbb{E}_X[\mathbb{E}[\theta|X]^2] = \mathbb{E}[\theta^2] - \mathbb{E}_X[\mathbb{E}[\theta|X]^2]\\ &= \mathbb{E}[\theta^2] - \sum_x \frac{e_\pi(x)^2}{m_\pi(x)}\end{aligned}$$

where $m_\pi(x) = \int p(x|\theta)d\pi(\theta)$ and $e_\pi(x) = \int \theta p(x|\theta)d\pi(\theta)$ are the PMF and posterior mean of x, respectively.

Algorithm 1 Pseudocode that approximates $\hat{\theta}_{\mathsf{Rob},d,\theta_{\max}}$ using a transformer.

```
Input: Inputs $x_1, \cdots, x_n$, $\theta_{\max}$, $d$.
Define: $d_k = d + 1$, $n_{head} = 1$.
Define: $D = \max\{100, d_k^2\}$.
Define: $W_Q = I_{d_k}$, $W_V = \mathrm{diag}(1, 1, \cdots, 1, 0), W_K$.
for $i = 1$ to $d + 1$ do
    $W_k[i, i] = D$
end for
for $i = 1$ to $d$ do
    $W_k[i, i + 1] = D + \sqrt{d_k} \log i$
end for
Define: $\mathrm{AttnLayer} = \mathrm{Attn}(W_Q, W_K, W_V)$.
Define: $Z = \mathrm{AttnLayer}(Y, Y, Y)$.
$Z' = \mathsf{ReLU}(Y + Z - 1)$.
$Z_1 = \mathrm{rowsum}(Z')$.
return $\min\{\frac{1}{Z_1} - 1, M\}$.
```

Now denote $e_{\theta_0}(x) = \theta_0 p(x|\theta_0)$ and $m_{\theta_0}(x) = p(x|\theta_0)$. Denote also the difference $d(x) \triangleq m_{\theta_0}(x) - m_{\pi_!}(x)$ and $k(x) \triangleq e_{\theta_0}(x) - e_{\pi_!}(x)$. Then

$$\begin{aligned}
&\mathsf{mmse}(\pi_\epsilon) \\
&= \mathbb{E}_{\pi_!}[\theta^2] + \epsilon(\theta_0^2 - \mathbb{E}_{\pi_!}[\theta^2]) - \sum_x \frac{(e_{\pi_!}(x) + \epsilon k(x))^2}{m_{\pi_!}(x) + \epsilon d(x)}
\end{aligned}$$

which means the derivative when evaluated at 0:

$$\begin{aligned}
0 &\geq \frac{\partial}{\partial \epsilon} \mathsf{mmse}(\pi_\epsilon)|_{\epsilon=0} \\
&= \theta_0^2 - \mathbb{E}_{\pi_!}[\theta^2] - \sum_x \frac{2m_{\pi_!}(x)e_{\pi_!}(x)k(x) - e_{\pi_!}(x)^2 d(x)}{m_{\pi_!}(x)^2} \\
&= \theta_0^2 - \sum_x \frac{2m_{\pi_!}(x)e_{\pi_!}(x)e_{\theta_0}(x) - e_{\pi_!}(x)^2 m_{\theta_0}(x)}{m_{\pi_!}(x)^2} \\
&\quad - \mathsf{mmse}(\pi_!) \\
&= \theta_0^2 - 2\sum_x e_{\theta_0}(x) \frac{e_{\pi_!}(x)}{m_{\pi_!}(x)} + \sum_x m_{\theta_0}(x) \left(\frac{e_{\pi_!}(x)}{m_{\pi_!}(x)} \right)^2 \\
&\quad - \mathsf{mmse}(\pi_!) \\
&= \mathsf{mse}_{\delta_\theta}(f_{\pi!}) - \mathsf{mmse}(\pi_!)
\end{aligned}$$

where the last equality follows from that $\frac{e_{\pi_!}(x)}{m_{\pi_!}(x)} = f_{\pi_!}(x)$. Therefore the conclusion follows. □

C Pseudocode on Robbins Approximation via Transformers

We present a pseudocode in Algorithm 1 on how a transformer can be set up to approximate Robbins, using a formulation that closely mimics the PyTorch module. All vectors and matrices use 1-indexing. Note that the attention output is $Z = \mathrm{Softmax}(\frac{YW_Q W_K^T Y^T}{\sqrt{d_k}})YW_V$.

D Further Analysis on Experimental Results

D.1 Synthetic Experiments

We recall that our synthetic experiments are measuring regret w.r.t. sequence length for both the neural and worst-prior. In the main section, we show a plot of how the regret decreases with sequence length; here, we provide a more comprehensive result on the Plackett-Luce rankings in Table 5, along with the p-value by pairwise t-test of T18 and T24 against relevant classical baselines, as per Table 6 and Table 7. From the p-value we conclude that the transformers outperform other baselines by a significant margin on various experiments. (except in a handful of cases).

Table 5: Plackett-Luce coefficients of estimators' regrets on synthetic experiments. The coefficient of MLE is set to 0 throughout.

Experiments	GS	Robbins	ERM	NPMLE	T18	T24
Neural-128	-0.003	-3.196	0.965	4.310	7.497	**7.696**
Neural-256	-0.023	-3.090	1.678	4.885	7.624	**8.002**
Neural-512*	-0.044	-3.016	2.421	5.534	7.646	**8.084**
Neural-1024	-0.066	-2.930	3.032	6.197	7.579	**7.983**
Neural-2048	-0.092	-2.813	3.430	6.806	7.455	**7.816**
WP-128	-	-4.925	-2.434	2.476	**7.416**	4.945
WP-256	-	-2.470	2.470	4.943	**9.878**	7.412
WP-512	-	-2.466	2.463	4.924	**9.842**	7.385
WP-1024	-	-2.738	2.735	8.543	**8.664**	5.476
WP-2048	-	-2.468	2.470	**9.863**	7.405	4.938
Multn-512	0.505	-2.664	2.239	4.877	**9.686**	7.339
Multn-1024	0.463	-2.635	3.328	5.764	**9.615**	8.240
Multn-2048	0.471	-2.728	3.432	5.963	**8.971**	8.811

Table 6: $\mathbb{P}[\mathsf{Regret}(\text{T18}) > \mathsf{Regret}(\text{Classical})]$ obtained via paired t-test.

Experiments	MLE	GS	Robbins	ERM	NPMLE
Neural-128	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-256	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-512*	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-1024	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-2048	< 1e-100	<1e-100	< 1e-100	< 1e-100	0.987
WP-128	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-256	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-512	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-1024	< 1e-100	-	< 1e-100	< 1e-100	7.13e-04
WP-2048	< 1e-100	-	< 1e-100	< 1e-100	>1 - 1e-100
Multn-512	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Multn-1024	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Multn-2048	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100

Table 7: $\mathbb{P}[\mathsf{Regret}(\text{T24}) > \mathsf{Regret}(\text{Classical})]$ obtained via paired t-test.

Experiments	MLE	GS	Robbins	ERM	NPMLE
Neural-128	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-256	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-512*	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-1024	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Neural-2048	< 1e-100	<1e-100	< 1e-100	< 1e-100	0.273
WP-128	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-256	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-512	< 1e-100	-	< 1e-100	< 1e-100	< 1e-100
WP-1024	< 1e-100	-	< 1e-100	< 1e-100	>1 - 1e-100
WP-2048	< 1e-100	-	< 1e-100	< 1e-100	>1 - 1e-100
Multn-512	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Multn-1024	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100
Multn-2048	< 1e-100	<1e-100	< 1e-100	< 1e-100	< 1e-100

D.2 Real Data Experiments

In the main section, we focused on reporting the RMSE. We supplement our finding on RMSE by showing a Plackett Luce ELO coefficient of RMSEs of the estimators in Table 8, which shows that the transformers are consistently ranked at the top.

Here, we also study what happens if we compare the Mean Absolute Error (MAE) of various estimators on each datapoint. Apart from the average percentage improvement of each algorithm over the MLE as per Table 9, we also display the p-values based on paired t-test of transformers vs other algorithms in Table 10, and Table 11. In Fig. 11, Fig. 12, Fig. 13 and Fig. 14, we also include the violin plots.

Table 8: Plackett-Luce coefficients of estimators' RMSE on real datasets. The coefficient of MLE is set to 0 throughout.

Dataset	Robbins	ERM	NPMLE	T18	T24
NHL	-2.536	1.458	3.252	**3.756**	3.587
NHL (defender)	-1.730	1.577	3.973	5.366	**5.636**
NHL (center)	-2.739	0.399	2.111	3.118	**3.408**
NHL (winger)	-3.350	0.674	2.271	**3.004**	2.756
MLB (batting)	-2.981	2.991	5.145	6.221	**8.575**
MLB (pitching)	-3.217	3.225	5.916	6.696	**7.689**
BookCorpusOpen	0.024	1.547	2.341	2.012	**2.698**

Table 9: 95% confidence interval of the percentage improvement of MAE by each algorithm over MLE.

Dataset	Robbins	ERM	NPMLE	T18	T24
Hockey (all)	-20.14 ± 4.44	-0.20 ± 0.60	**0.90 ± 0.58**	0.76 ± 0.58	0.77 ± 0.59
Hockey (defender)	-13.38 ± 4.24	1.44 ± 1.16	3.26 ± 1.07	**3.66 ± 1.23**	3.62 ± 1.22
Hockey (center)	-41.43 ± 9.21	-0.65 ± 0.86	2.26 ± 0.82	**2.87 ± 0.86**	2.76 ± 0.86
Hockey (winger)	-32.43 ± 7.23	-0.12 ± 0.78	1.43 ± 0.63	**1.77 ± 0.67**	1.68 ± 0.67
Baseball (batting)	-25.97 ± 3.81	3.50 ± 0.32	5.22 ± 0.36	**5.60 ± 0.34**	5.53 ± 0.36
Baseball (pitching)	-16.74 ± 2.19	3.45 ± 0.46	5.65 ± 0.48	5.60 ± 0.47	**5.70 ± 0.44**
BookCorpusOpen	28.05 ± 0.14	29.54 ± 0.10	29.65 ± 0.10	**31.70 ± 0.15**	27.85 ± 0.15

Table 10: $\mathbb{P}[\text{MAE(Transformers)} > \text{MAE(Classical)}]$ obtained via paired t-test.

	T18				T24			
Dataset	MLE	Robbins	ERM	NPMLE	MLE	Robbins	ERM	NPMLE
NHL	0.0103	6.69e-10	9.32e-05	0.868	0.0111	7.95e-10	1.33e-04	0.887
NHL (defender)	9.42e-07	6.53e-10	1.54e-05	1.78e-03	9.03e-07	6.22e-10	4.45e-05	0.013
NHL (center)	5.97e-07	3.04e-10	3.27e-08	2.48e-03	1.02e-06	3.98e-10	1.23e-07	8.20e-03
NHL (winger)	7.33e-06	7.49e-10	1.13e-05	0.0173	1.35e-05	7.98e-10	2.75e-05	0.0518
MLB batting	1.63e-24	6.01e-16	5.06e-12	6.57e-07	3.68e-24	6.74e-16	1.49e-11	3.04e-07
MLB pitching	5.67e-21	2.76e-16	6.36e-12	0.780	7.27e-22	2.38e-16	7.12e-13	0.203
BookCorpusOpen	<1e-100	<1e-100	5.31e-26	3.62e-23	<1e-100	9.5e-69	1-6.32e-10	1-1.87e-12

Table 11: Plackett-Luce coefficients of estimators' MAE on real datasets. The coefficient of MLE is set to 0 throughout.

Dataset	Robbins	ERM	NPMLE	T18	T24
NHL	-2.928	-0.023	**1.769**	1.490	1.435
NHL (defender)	-2.182	0.795	2.170	**2.674**	2.667
NHL (center)	-3.201	-0.472	1.678	**2.617**	2.525
NHL (winger)	-2.820	0.196	1.464	**2.302**	1.916
MLB batting	-3.076	3.080	5.572	**7.968**	7.391
MLB pitching	-3.332	3.331	6.677	6.253	**7.146**
BookCorpusOpen	4.325	4.896	5.067	**5.319**	4.704

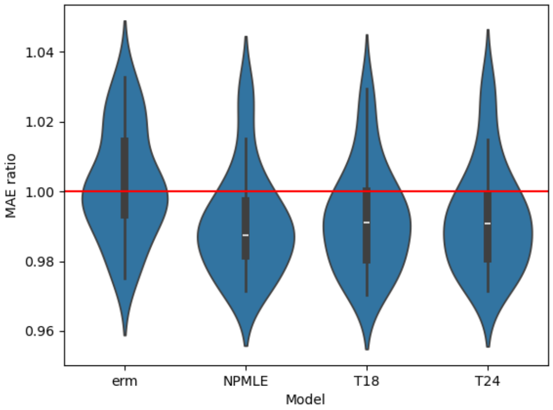


Figure 11: Violin plots of MAE ratio achieved by multiple estimators over MLE on NHL.

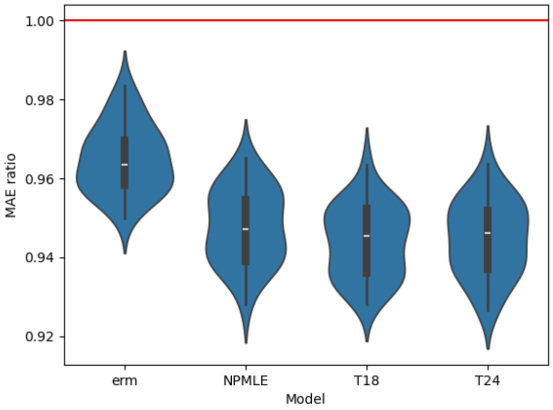


Figure 12: Violin plots of MAE ratio achieved by multiple estimators over MLE on MLB batting.

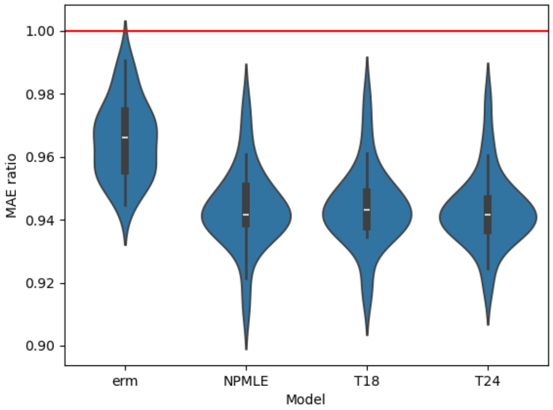


Figure 13: Violin plots of MAE ratio achieved by multiple estimators over MLE on MLB pitching.

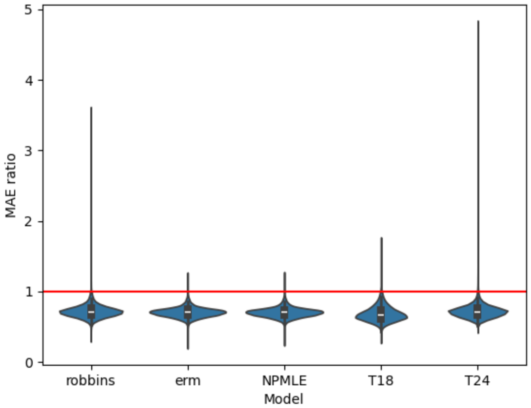


Figure 14: Violin plots of MAE ratio achieved by multiple estimators over MLE on BookCorpusOpen.