Can Transformer Decoders do Model Selection?

Edward Ri

April 5th, 2023

1 Introduction

Recent literature has demonstrated the capacity of Transformer decoder models to perform in-context learning in linear regression models. These models are trained on input-output pairs generated by randomly chosen weight vectors, and studies have shown that they can achieve ordinary least squares (OLS) loss in both noiseless and noisy linear regression learning settings (Garg et al., 2023; Akyürek et al., 2022).

This short report investigates whether the same model architecture is capable of model selection among similar tasks. Specifically, we explore the scenario where the model is trained on data generated from T types of models and is given a set of data points with labels produced by one of the possible models at testing time to use as in-context examples along an unlabeled input. The model must then generate an output for the input based on the provided examples.

We focus on the simple setting where $|T| = 2$, consisting of the noiseless and noisy linear regression tasks. We test various parameters to determine when the ability to perform model selection is acquired.

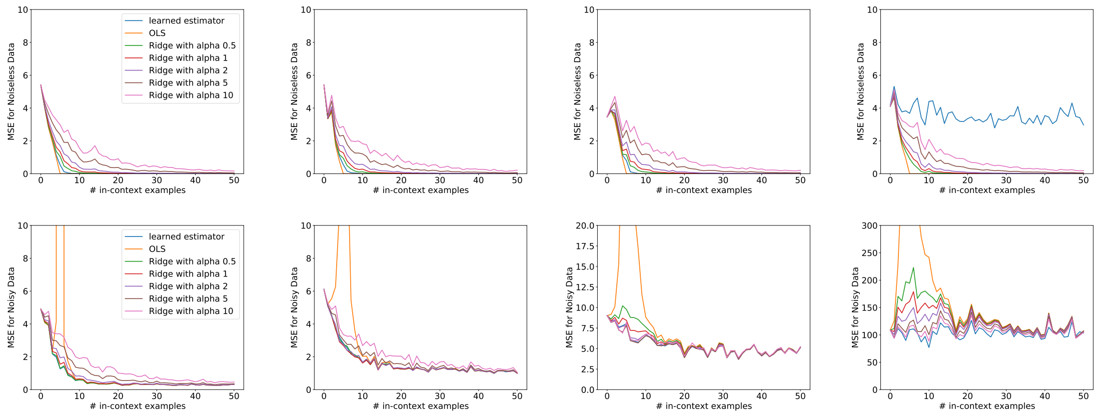
2 Problem Setup

We adopt the procedure in Garg et al. (2023) and sample a set of input points $x_1, \dots, x_n$ and a random unit vector w of dimension d from an isotropic Gaussian distribution, which is used as the weight vector. The labels for each input data point are generated by computing $y_i = w^\intercal x_i$ for all $i \in [n]$. The Transformer architecture takes the concatenated input sequence $(x_1, y_1)|(x_2, y_2)|, ...|(x_n, y_n)|(x_{n+1})$ and is trained to predict the value of y_{n+1}.

During testing, a new weight vector w and a set of input points $x_1, \dots, x_n$ are randomly selected to construct the in-context examples. The model takes the concatenated input sequence $(x_1, y_1)|(x_2, y_2)|, ...|(x_n, y_n)|(x_{n+1})$ as input and predicts the value of y_{n+1}.

Model	**Layers**	**Emb. Space**	**Heads**
`Tiny`	3	64	2
`Small`	6	128	4
`Standard`	12	256	8

Table 1: Model architecture details.



(a) $\epsilon = 0.5$ (b) $\epsilon = 1$ (c) $\epsilon = 2$ (d) $\epsilon = 10$

Figure 1: Experiments with $d = 5, n = 51$ with curriculum learning. Top plot indicates MSE loss for noiseless case, bottom plot indicates MSE loss for noisy case where noisy data is perturbed with Gaussian noise from $N(0, \epsilon)$.

A list of available models is provided in Table 1. For all experiments run in this report, we use the `Standard` model where Transformer decoders are stacked to construct the models.

3 Experiments

To validate the capability of the model for model selection in linear regression, we employ a training approach using a mixture of noisy and noiseless data.

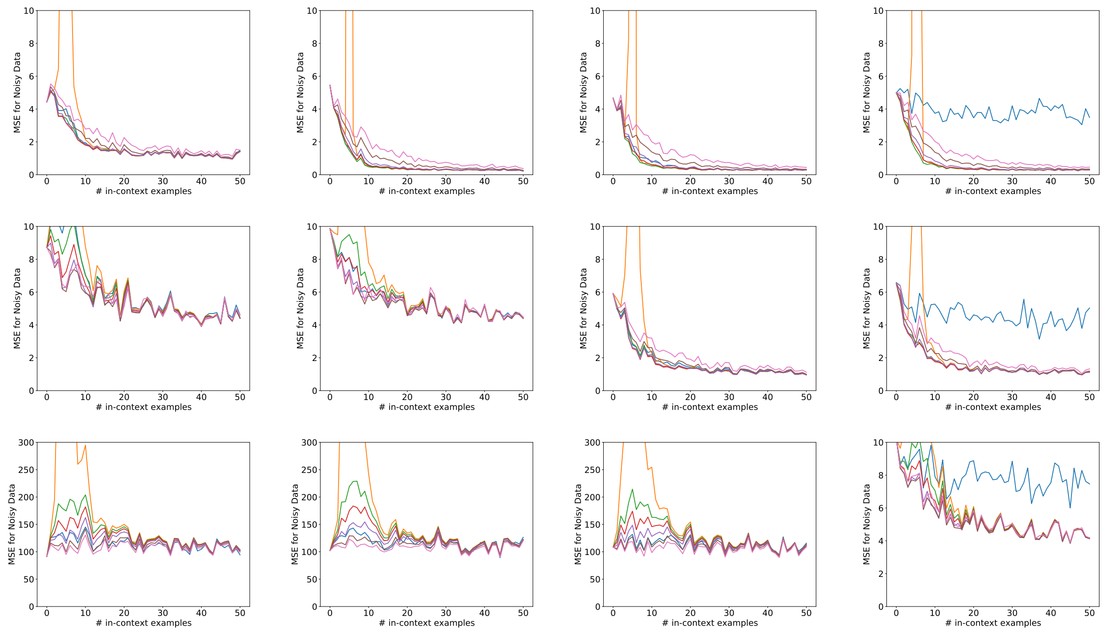
At each time step, we randomly sample a weight vector and input data to generate noiseless training data for the model. We then introduce Gaussian noise $N(0, \epsilon)$ to the output data y_i with a probability of $\frac{1}{2}$, where ϵ is predefined by the user.

3.1 Validation on Training Distribution

Procedure We evaluate the performance of our model by generating training data $(x_1, y_1), \dots, (x_n, y_n)$ in a similar manner to the training phase, where we randomly sample a weight vector and input data to pass to the model.

We test the model on two cases: when the input data x_i is not perturbed by noise, and when the data is perturbed by Gaussian noise with variance ϵ, i.e $x_i = x_i + \delta$ where $\delta \sim N(0, \epsilon)$. The model is then fed the training data $(x_1, y_1)| \dots |(x_k)$ and we compute the mean squared error between the predicted value $\hat{y_k}$ and the ground truth value y_k.

Note that during testing, the model is provided with the *correct* y_k value instead of using the output of the model at the $(k + 1)$th time step (also known as *teacher forcing*), i.e. the sequence $(x_1, y_1)| \dots |(x_k, y_k)|(x_{k+1})$ is used as input at the next time step.



(a) $\epsilon = 0.5$, $\epsilon' = 1, 2, 10$ (b) $\epsilon = 1$, $\epsilon' = 0.5, 2, 10$ (c) $\epsilon = 2$, $\epsilon' = 0.5, 1, 10$ (d) $\epsilon = 10$, $\epsilon' = 0.5, 1, 2$

Figure 2: Experiments with distribution shifts of the input data distribution. ϵ denotes the noise rate the model encounters during training, ϵ' denotes the noise rate of the test data. Note that all experiments use noisy data. Graph colors inherit from Figure 1.

Results The experimental results for $d = 5$ and $n = 51$, where n represents the number of in-context examples, are presented in Figure 1. The first row of Figure 1 shows the loss of the trained model compared to that of OLS and Ridge Regression, which are parameterized by λ with $\lambda = \epsilon$.

We observe that the learned estimator closely follows the error curves of Ridge regression with ϵ as lambda for small enough values of ϵ. However, for significantly large values of ϵ, the model fails to match the error curves of both OLS and Ridge regression.

3.2 Validation on Distribution Shifts

Testing Procedure In contrast to the previous section, we also test the model's performance for the noisy input data case where the test noise rate ϵ' differs from the noise rate ϵ as what the model was trained on.

Results Results are given in Figure 2, where the test noise rates are ordered in increasing order by row. For all experiments, we see that when the model is tested on a different noise rate than what it encounters during training, the model's error curves does not necessarily match that of ridge regression with $\lambda = \epsilon$.

In Figure 2b where $\epsilon = 1, \epsilon' = 0.5, 1$, we see that the model's error curves match that of $\epsilon = 1$ but does not match when $\epsilon' = 10$.

We also inconsistently see cases where the model's curve matches that of ridge regression with a different value for λ. Namely, for the bottom column of Figure 2a, we see that the error curve of the model most closely matches that of $\lambda = 2$. Note that the MSE values are drastically larger than other cases for $\epsilon' = 10$, and thus the two graphs are not necessarily closely matching in comparison to other error rate cases.

For $\epsilon = 10$ (Figure 2d), we see that the model's error curves hover above all other error curves, indicating the deficiency of training the model with a dramatically large noise rate.

4 Discussion

Is Curriculum Learning Necessary? Garg et al. (2023) empirically observed that the model takes a dramatically longer amount of time to train until the model's loss at test time matches OLS. We have utilized curriculum learning throughout the experimennt as it is unlikely that removing curriculum learning from the training procedure has any interesting effects on the model selection task.

Potentially Interesting Questions We list a number of potential directions for future research:

- Taking inspiration from Akyürek et al. (2022), can we mathematically show that it is possible to implement some kind of "model selection" algorithm using the Transformer decoder architecture?
- Is there a lower bound on the architecture complexity for the emergence of model selection as a function of the dimensions and number of in-context examples?

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models, 2022.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023.