A Closer Look at In-Context Learning under Distribution Shifts

Kartik Ahuja [1] **David Lopez-Paz** [1]

Abstract

In-context learning, a capability that enables a model to learn from input examples on the fly without necessitating weight updates, is a defining characteristic of large language models. In this work, we follow the setting proposed in (Garg et al., 2022) to better understand the generality and limitations of in-context learning from the lens of the simple yet fundamental task of linear regression. The key question we aim to address is: Are transformers more adept than some natural and simpler architectures at performing in-context learning under varying distribution shifts? To compare transformers, we propose to use a simple architecture based on set-based Multi-Layer Perceptrons (MLPs). We find that both transformers and set-based MLPs exhibit in-context learning under in-distribution evaluations, but transformers more closely emulate the performance of ordinary least squares (OLS). Transformers also display better resilience to mild distribution shifts, where set-based MLPs falter. However, under severe distribution shifts, both models' in-context learning abilities diminish.

1. Introduction

Transformers (Vaswani et al., 2017) form the backbone of modern large language models (LLMs) including the likes of GPT-3 (Brown et al., 2020) and GPT-4 (OpenAI, 2023). These LLMs demonstrate remarkable capabilities, such as in-context learning and natural language-based algorithmic reasoning. However, we are only beginning to understand the origins, limitations, and generality of these capabilities, which is essential for developing safe and reliable LLMs.

In-context learning (ICL) refers to a model's capability to acquire knowledge on the fly from examples provided at test time without requiring any weight updates. This ability is especially useful when the model has to adapt to new tasks from a few demonstrations in the test prompt, for example, adapting a model to drive in a new region with few demonstrations. Understanding ICL for LLMs such as GPT-3 trained on raw text data is particularly challenging. In (Garg et al., 2022), the authors propose an insightful training setup, which abstracts away the raw nature of text data. In their work, transformer models from GPT-2 family are trained on prompts comprising of input, label demonstrations and shown to emulate the ordinary least squares (OLS) algorithm. Certain natural questions arise at this point. What specifics of the transformer are responsible for the emergence of this behvavior? Can simpler architectures exhibit the same capabilities? How resilient is ICL to distribution shifts? These are the questions that motivate our work.

To compare with transformers, we propose a natural baseline that is based on set-based MLPs (Zaheer et al., 2017; Lopez-Paz et al., 2017) that exploit the permutation-invariant nature of the task. Depending on the distribution of test prompts, we categorize in-context learning into in-distribution ICL (ID-ICL) and out-of-distribution ICL (OOD-ICL). Under ID-ICL, the train distribution of the prompt is identical to the test distribution of the prompt. Under OOD-ICL, the test distribution of prompt sequence is different from the train distribution. When evaluating OOD-ICL, we are particularly interested in the case when the test distribution of prompts is centered on the tail of the train distribution of prompts. We summarize our key contributions below.

- First, we derive conditions under which the the optimal model that predicts the label for the current query based on the prompt coincide with the OLS or ridge regression. These are based on known arguments, yet it is important to provide them for completeness.
- Despite set-based MLPs being particularly suited for this permutation-invariant task, we find that transformers (GPT-2 family) exhibit better ID-ICL abilities.
- Under mild distribution shifts, we find that transformers degrade more gracefully than set-based MLPs. Under more severe distribution shifts, both transformers and set-based MLPs do not exhibit ICL abilities.
- ID-ICL performance is not predictive of OOD-ICL performance for both architecture choices.

[*]Equal contribution [1]FAIR, Meta AI. Correspondence to: Kartik Ahuja <kartikahuja@meta.com>.

Work presented at the ES-FoMo Workshop at ICML 2023, Honolulu, Hawaii, USA.

Moving forward, several questions need to be answered. Why are transformers better than set-based MLPs at ICL? How can we improve the OOD-ICL abilities of these architectures?

2. Related Works

Recent studies have offered intriguing insights into in-context learning (ICL). Olsson et al. (2022) propose that the formation of "induction heads", which allow models to copy in-context information, is key to ICL. Building on (Garg et al., 2022)'s work, several researchers (Akyürek et al., 2022; von Oswald et al., 2022; Dai et al., 2022) demonstrated that transformer model's ability to implicitly execute gradient descent steps during inference could also be central to ICL, supporting their claims with empirical evidence. Li et al. (2023) explore this setup further by analyzing generalization bounds for the learnability of algorithms. Lastly, Xie et al. (2021) focus on data sampled from hidden Markov model and interpret in-context learning through the lens of implicit Bayesian inference. They go on to provide conditions under which models can perform ICL even when prompts have low probability under the training distribution.

Chan et al. (2022) studied the impact of inductive bias of pretraining the model on ICL. The authors showed that pretrained transformers exhibit rule-based generalization, while those trained from scratch use exemplar-based generalization, i.e., leverage information from the examples provided in-context to carry out ICL. Kirsch et al. (2022b) find that among factors determining the inductive bias of the model, state-size is a more crucial parameter than the model size for ICL abilities. More recently, Wei et al. (2023) showed that model size can be a crucial parameter as well. In particular, they show that sufficiently large models such as PaLM-540B are capable of overriding semantic priors if needed, while smaller counterparts are unable to do so.

3. In-context Learning under Distribution Shifts

We start with some standard notation. Inputs and labels are denoted as $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$ respectively. Each prompt p is a sequence of independent and identically distributed (i.i.d.) input, label pairs, denoted as $p = \{(x_i, y_i)\}_{i=1}^k$. Each prompt p is sampled independently as follows

$$\begin{aligned} &f \sim \mathbb{P}_f, \\ &x_i \sim \mathbb{P}_x, \varepsilon_i \in \mathbb{P}_\varepsilon, \forall i \in \{1, \cdots, k\}, \\ &y_i \leftarrow f(x_i) + \varepsilon_i, \forall i \in \{1, \cdots, k\}, \end{aligned} \tag{1}$$

where the labeling function f, which is fixed for the entire prompt p, is sampled from a distribution $\mathbb{P}_f$, inputs x_i are sampled independently from $\mathbb{P}_x$, y_i is generated by adding some noise ε_i to the labeling function's output $f(x_i)$. For the prompt p, we define its prefix as $p_j = ((x_1, y_1), (x_2, y_2), \cdots, x_j)$, where $j \in \{1, \cdots, k\}$. Define the support of prefix p_j as $\mathcal{P}_j$.

Define the risk for model M as $R(M) = \sum_{j=1}^k \mathbb{E}\big[\ell\big(M(p_j), y_j\big)\big]$, where ℓ is the loss, $M(p_j)$ looks at the prefixes p_j and makes the prediction, the loss is computed w.r.t the true label y_j, $\mathbb{E}[\cdot]$ is the expectation over the joint distribution of (p_j, y_j). We want to find a model that minimizes the risk $R(M)$ i.e.,

$$M^* \in \arg\min_M R(M) \tag{2}$$

For the results to follow, we make some standard regularity assumptions that we state as follows. The probability measure associated with p_j is absolutely continuous w.r.t Lebesgue measure. The conditional expectation and variance exists, i.e., $|\mathbb{E}[y_j|p_j]| < \infty$ and $\mathsf{Var}[y_j|p_j] < \infty$ for all $p_j \in \mathcal{P}_j$.

Lemma 3.1. *If ℓ is the square loss, then the solution to equation (2) satisfies, $M^*(p_j) = \mathbb{E}[y_j|p_j]$, almost everywhere in $\mathcal{P}_j$, $\forall j \in \{1, \cdots, k\}$.*

While the above lemma is stated for square loss, an equivalent statement holds for cross-entropy loss. We now turn to our case study, i.e., linear labeling functions f. Each prompt p is sampled as follows

$$\begin{aligned} &\beta \sim \mathcal{N}(0, \Sigma), \text{where } \Sigma \in \mathbb{R}^{d\times d} \text{ is invertible} \\ &x_i \sim \mathbb{P}_x, \varepsilon_i \sim \mathcal{N}(0, \sigma^2), \forall i \in \{1, \cdots, k\} \\ &y_i \leftarrow \beta^\top x_i + \varepsilon_i, \forall i \in \{1, \cdots, k\} \end{aligned} \tag{3}$$

where β is drawn from a normal distribution with mean zero and covariance Σ and noise ε_i is sampled from a normal distribution with mean zero and variance σ^2. We break down prefix p_j into a matrix $\boldsymbol{X}_j \in \mathbb{R}^{(j-1)\times d}$ and vector $\boldsymbol{y}_j \in \mathbb{R}^{j-1}$ that stacks the first $j-1$ x_i's and y_i's observed in the prompt up to query x_j. The tuple $(\boldsymbol{X}_j, \boldsymbol{y}_j, x_j)$ captures all the relevant information from p_j for predicting y_j. Since p_1 has no inputs to look at in the past, we set $\boldsymbol{X}_1, \boldsymbol{y}_1$ to zero. To better understand the notation, consider the following example, $p = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$. Prefix $p_3 = \{(x_1, y_1), (x_2, y_2), x_3\}$, $\boldsymbol{X}_3 = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, $\boldsymbol{y}_3 = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$. Next, we derive the optimal models $M^*(p_j)$ for the data distribution in equation (3). The theorems derived below follows from standard results on linear regression (See Dicker (2016); Richards et al. (2021)). We still state and derive these for completeness.

Theorem 3.2. *If ℓ is the square loss and prompt generation follows equation (3), then the optimal model from equation (2) satisfies,*

$$M^*(p_j) = x_j^\top (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j, \forall j \in \{1, \cdots, k\}$.

If Σ is identity, then the above solution coincides with ridge regression (Hoerl & Kennard, 1970) using σ^2 as the ridge penalty. We now study the noiseless setting. To analyze the noiseless case, we will look at the ridge solutions in the limit of σ going to zero.

Theorem 3.3. *If ℓ is the square loss and prompt generation follows equation (3) with Σ as identity, [1] then in the limit of $\sigma \to 0$ the optimal model from equation (2) satisfies*

$$M^*(p_j) = x_j^\top \boldsymbol{X}_j^+ \boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j, \forall j \in \{1, \cdots, k\}$, where $\boldsymbol{X}_j^+$ is the Moore-Penrose pseudo-inverse of $\boldsymbol{X}_j$.

In the above results (Lemma 3.1, Theorem 3.2, and Theorem 3.3) we do not use the fact that inputs x_i's are drawn independently. In Theorem 3.2, and Theorem 3.3, we assumed that β is drawn from a normal distribution. For distributions beyond normal, we now argue that if we restrict the search space of models, then the same results continue to hold.

Constraint 3.4. $M(p_j) = x_j^\top m(\boldsymbol{X}_j) \boldsymbol{y}_j$.

The above constraint restricts the model to be linear in test query and also to be linear in the label seen up to that point. We do not impose any restrictions on $m(\cdot)$. In the absence of this constraint, the risk $R(M)$ depends on moments beyond the second order moments of the distribution of β. Thus the optimal model in the absence of this constraint may not coincide with OLS or ridge regression.

Theorem 3.5. *Suppose ℓ is the square loss, β's and x_i's are drawn from an arbitrary distribution with a finite mean and invertible covariance, rest of the prompt generation follows equation (3). In this setting, the solution to equation (2) under Constraint 3.4 satisfies*

$$M^*(p_j) = x_j^\top (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j$, $\forall j \in \{1, \cdots, k\}$.

So far, we have characterized different conditions under which the optimal model emulates the OLS or the ridge regression on the support of training distribution of the prompts. The study by Garg et al. (2022) demonstrated that transformers, when trained with sufficient data, can emulate OLS regression. Theorem 3.2, 3.3 suggest that sufficiently high capacity models (that can handle input data of varying lengths) trained on sufficient amount of data should behave as well as transformers on the prompts sampled from the same distribution as the train distribution. We test this hypothesis in the experiments section. Outside the support of the training distribution of prompts, performance is not guaranteed to be good, and it depends on the inductive biases – architecture, optimizer, and the loss function. Our experiments will examine the bias from the architecture. We now propose a natural architecture for the task in question.

A natural baseline for the above task We revisit the data generation in equation (1) and parametrize the labeling function. Say the labeling process now is $y_i \leftarrow f(x_i, \beta) + \varepsilon_i$, where β is sampled from some distribution. $\mathbb{E}[y_i | x_i, \beta] = f(x_i, \beta)$. Our model will first estimate β from the given set of samples $\boldsymbol{X}_j, \boldsymbol{y}_j$. The estimation of β does not depend on the order of inputs and thus estimation should be invariant w.r.t. to the order of inputs. Further, we want to work with architectures that are capable of handling inputs of variable length. For this purpose, the most natural architecture are the ones that accept sets as inputs. We revisit the Theorem 2 in (Zaheer et al., 2017). The theorem states

Theorem. *(Zaheer et al., 2017) A function operating on a set $\mathcal{A}$ having elements from a countable universe is a valid set function iff it can be expressed as $\rho\big(\sum_{a_i \in \mathcal{A}} \phi(a_i)\big)$.*

The aforementioned theorem is stated for elements from a countable universe, with its extension to uncountable sets provided in (Zaheer et al., 2017), albeit for fixed-length sets. Since functions of the form $\rho\big(\sum_{a_i \in \mathcal{A}} \phi(a_i)\big)$ are uninversal representers of set-based functions we use them as the basis for our architecture. We pick both ρ and ϕ as Multilayer Perceptrons (MLPs), and we use these to estimate the parameter β. The output from these MLPs is then input into another MLP together with the query x_j. The final architecture takes the form $\psi\left(\rho\big(\frac{1}{j-1}\sum_{i=1}^{j-1} \phi(x_i, y_i)\big), x_j\right)$, where (x_i, y_i) are input, label pairs seen up to x_j. To manage sequences of variable length, we incorporate a normalization term $\frac{1}{j-1}$. Consider the noisy label scenario that we studied in Theorem 3.3, where the optimal model is defined by $x_j^\top (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j$. Here, $\rho\big(\frac{1}{j-1}\sum_{i=1}^{j-1} \phi(x_i, y_i)\big)$ aims to output the best estimate for β, which is $\hat{\beta}(\boldsymbol{X}_j, \boldsymbol{y}_j) = (\boldsymbol{X}_j^\top \boldsymbol{X} j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j$; note how $\hat{\beta}(\boldsymbol{X}_j, \boldsymbol{y}_j)$ is permutation-invariant. As per (Zaheer et al., 2017), sufficiently expressive ρ and ϕ should be capable of expressing $\hat{\beta}(\boldsymbol{X}_j, \boldsymbol{y}_j)$. The final MLP, ψ, must approximate a linear map. Next, we delve into the distribution shifts we consider and their underlying rationale.

Distribution shifts for ICL. In both regression and classification problems, the concept of covariate shift (Shi-

[1] If Σ is not identity, then the limit may or may not coincide with OLS; see the Appendix for further discussion.

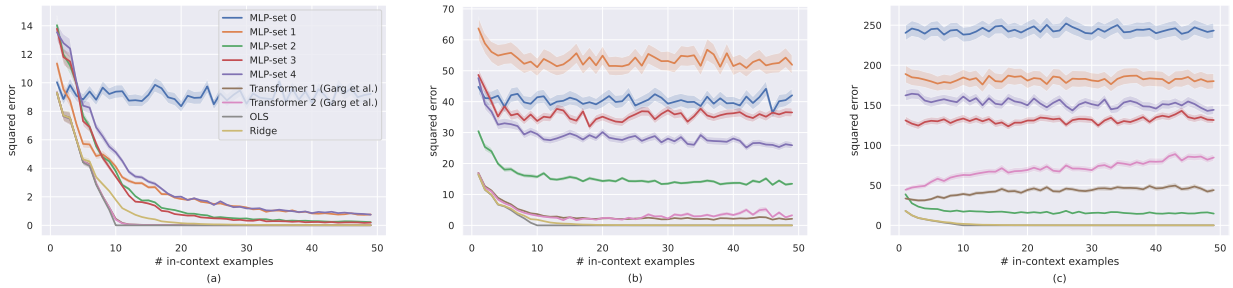

Figure 1. Comparison of MLP-set and transformers for noiseless setting, i.e., $\sigma = 0$. a) ID-ICL ($\mu = 0$), b) OOD-ICL (Mild distribution shift with $\mu = 2 \cdot \mathbf{1}$), c) OOD-ICL (Severe distribution shift with $\mu = 4 \cdot \mathbf{1}$).

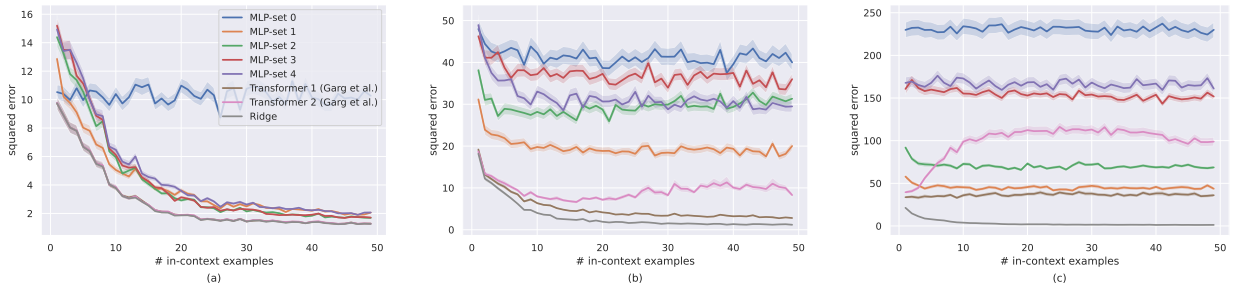

Figure 2. Comparison of MLP-set and transformers for noisy setting, i.e., $\sigma = 1$. a) ID-ICL ($\mu = 0$), b) OOD-ICL (Mild distribution shift with $\mu = 2 \cdot \mathbf{1}$), c) OOD-ICL (Severe distribution shift with $\mu = 4 \cdot \mathbf{1}$).

modaira, 2000) is well-understood. Covariate shift refers to the situation where the distribution of the input features, denoted as $\mathbb{P}_x$, changes between training and testing phases, but the conditional distribution of the target variable given the features remains invariant. This idea can be applied to the prompts p. When the distribution over prompts changes, but the conditional distribution of the target variable (or response) given the prompt remains invariant, this is referred to as "covariate shift over prompts". This is a particularly important setting to test, as it helps us understand the model's ability to learn from novel types of prompts or demonstrations at test time.

Consider two examples that leverage equation (3) as the underlying data generation process. Suppose at train time, we generate prompt sequences with inputs x_i's that are mostly positive and then test on prompts comprised of negative inputs. If between train and test we do not alter the label generation process, then this setting qualifies as covariate shift over prompts. On the other hand, consider the setting, where the only difference from train to test is that during label generation at test time is noisy. In this case, the prompt distribution changes but the conditional distribution of the target conditional on the prompt also changes ($\mathbb{E}[y|p]$ at train time is the OLS solution and at test time it is the ridge regression solution). As a result, this type of shift does not qualify as covariate shift over prompts. We want to remark that the difference between two models that perfectly minimize the expected loss in equation (2) is not apparent under all types of covariate shifts but those that put much more weight on input sequences that are very low probability at train time. This is one aspect in which our choice of distribution shifts differs from (Garg et al., 2022).

4. Experiments

In this section, we experiment with the set-based MLPs detailed earlier and transformers from (Garg et al., 2022). We generate data in line with the equation (3). The inputs $x_i's$ at train time are sampled from $\mathcal{N}(0, I_d)$, where I_d is the d dimensional identity matrix, and at test time they are sampled from $\mathcal{N}(\mu, I)$. In one case, we set $\mu = 2 \cdot \mathbf{1}$ and refer to it as a mild distribution shift, and in another case we set $\mu = 4 \cdot \mathbf{1}$ as severe distribution shift, where $\mathbf{1}$ is a d dimensional vector of all ones. The results are presented for $d = 10$. The covariance of β, i.e., Σ is identity. We present results for both noiseless labels and noisy labels with $\sigma^2 = 1$. For the set-based MLPs, which we refer to as MLP-set, we compare the performance of MLP-set under varying depths, $\{4, 5, 10, 17, 26\}$ (indexed from 0 to 4 in the increasing order of depth). The width was same for all the layers at 500. We trained the MLP-set model with the Adam optimizer and a learning rate of 0.001 except for the case of depth 26, where we had to lower the learning rate to 0.0001 to enable learning. We used ReLU activations and

batch norm between any two hidden layers. For training the transformer model, we adopt the same architecture used in (Garg et al., 2022), which belongs to the GPT-2 family, and we include performances at two depths - 12 (Transformer 1) and 16 (Transformer 2).

With this experimental setup we ask these key questions: existing works studying this ICL framework from (Garg et al., 2022) focused on transformers exhibiting this capabiltiy. Can this ability exist in other models such as the set-based MLPs? How do the two architectures differ under distribution shifts? In Figure 1, 2, we compare the two architectures for the noiseless and noisy setting respectively. We describe our key findings below

- We find that set-based MLPs exhibit ID-ICL capabilities but do not match the performance of transformers; see Figure 1a, 2a. This is inspite of choosing an architecture that is well suited for the task.
- Under mild distribution shifts; see Figure 1b, 2b, transformers exhibit a more graceful degradation as opposed set-based MLPs that become more erratic.
- Under more severe distribution shifts; see Figure 1c, 2c, both the transformers and the set-based MLPs do not exhibit OOD-ICL abilities.
- Finally, the ranking of ID-ICL performance of either the set-based MLPs or the transformers is not predictive of their OOD-ICL abilities.

5. Discussion

This research reveals that transformers outperform natural baselines in approximating OLS and ridge regression algorithms under mild distribution shifts. The question remains, why are transformers superior? Further investigation is required to theorize why transformers when optimized with familiar optimizers like stochastic gradient descent (SGD), can achieve better approximations of algorithms than set-based MLPs. Additionally, it's crucial to explore if these comparisons hold up for a broader set of algorithms (beyond OLS), architectures (beyond set-based MLPs, (Kirsch & Schmidhuber, 2021; Kirsch et al., 2022a)), and understand why. Some important steps towards these inquiries have been made by Liu et al. (2022).

References

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Albert, A. E. *Regression and the Moore-Penrose pseudoinverse*. Academic press, 1972.

Ash, R. B. and Doléans-Dade, C. A. *Probability and measure theory*. Academic press, 2000.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Chan, S. C., Dasgupta, I., Kim, J., Kumaran, D., Lampinen, A. K., and Hill, F. Transformers generalize differently from information stored in context vs in weights. *arXiv preprint arXiv:2210.05675*, 2022.

Dai, D., Sun, Y., Dong, L., Hao, Y., Sui, Z., and Wei, F. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*, 2022.

Dicker, L. H. Ridge regression and asymptotic minimax estimation over spheres of growing dimension. 2016.

Garg, S., Tsipras, D., Liang, P., and Valiant, G. What can transformers learn in-context? a case study of simple function classes. *arXiv preprint arXiv:2208.01066*, 2022.

Hoerl, A. E. and Kennard, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Kirsch, L. and Schmidhuber, J. Meta learning backpropagation and improving it. *Advances in Neural Information Processing Systems*, 34:14122–14134, 2021.

Kirsch, L., Flennerhag, S., van Hasselt, H., Friesen, A., Oh, J., and Chen, Y. Introducing symmetries to black box meta reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7202–7210, 2022a.

Kirsch, L., Harrison, J., Sohl-Dickstein, J., and Metz, L. General-purpose in-context learning by meta-learning transformers. *arXiv preprint arXiv:2212.04458*, 2022b.

Li, Y., Ildiz, M. E., Papailiopoulos, D., and Oymak, S. Transformers as algorithms: Generalization and stability in in-context learning. 2023.

Liu, B., Ash, J. T., Goel, S., Krishnamurthy, A., and Zhang, C. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

Lopez-Paz, D., Nishihara, R., Chintala, S., Scholkopf, B., and Bottou, L. Discovering causal signals in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6979–6987, 2017.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

OpenAI. Gpt-4 technical report. *arXiv*, 2023.

Richards, D., Mourtada, J., and Rosasco, L. Asymptotics of ridge (less) regression under general source condition. In *International Conference on Artificial Intelligence and Statistics*, pp. 3889–3897. PMLR, 2021.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent. *arXiv preprint arXiv:2212.07677*, 2022.

Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

A. Appendix

Lemma. *[Restatement of Lemma 3.1] If ℓ is the square loss, then the solution to equation (2) satisfies, $M^*(p_j) = \mathbb{E}[y_j|p_j]$, almost everywhere in $\mathcal{P}_j$, $\forall j \in \{1, \cdots, k\}$.*

Proof. We write

$$R(M) = \sum_{j=1}^{k} R_j(M),$$

where $R_j(M) = \mathbb{E}\Big[\ell\big(M(p_j), y_j\big)\Big]$. We simplify $R_j(M)$ below

$$\begin{aligned}
&R_j(M) \\
&= \mathbb{E}\Big[\ell\big(M(p_j), y_j\big)\Big] = \mathbb{E}_{p_j}\mathbb{E}_{y_j|p_j}\Big[\big(M(p_j) - y_j\big)^2\Big] \\
&= \mathbb{E}_{p_j}\mathbb{E}_{y_j|p_j}\Big[\big(M(p_j) - \mathbb{E}[y_j|p_j] + \mathbb{E}[y_j|p_j] - y_j\big)^2\Big] \\
&= \mathbb{E}_{p_j}\mathbb{E}_{y_j|p_j}\Big[\big(M(p_j) - \mathbb{E}[y_j|p_j]\big)^2\Big] + \mathbb{E}_{p_j}\mathbb{E}_{y_j|p_j}\Big[\big(y_j - \mathbb{E}[y_j|p_j]\big)^2\Big] + 2\mathbb{E}_{p_j}\mathbb{E}_{y_j|p_j}\Big[\big(M(p_j) - \mathbb{E}[y_j|p_j]\big)\big(y_j - \mathbb{E}[y_j|p_j]\big)\Big] \\
&= \mathbb{E}_{p_j}\Big[\big(M(p_j) - \mathbb{E}[y_j|p_j]\big)^2\Big] + \mathbb{E}_{p_j}\big[\mathsf{Var}[y_j|p_j]\big]
\end{aligned} \tag{4}$$

Observe that $R_j(M) \geq \mathbb{E}_{p_j}\big[\mathsf{Var}[y_j|p_j]\big]$ and thus $R(M) \geq \sum_{j=1}^{k}\mathbb{E}_{p_j}\big[\mathsf{Var}[y_j|p_j]\big]$. If M^* is a minimizer of $R(M)$, then it also has to minimize $R_j(M)$. If that were not the case, then M^* could be strictly improved by replacing M^* for the j^{th} query with the better model, thus leading to a contradiction. Consider the model $\tilde{M}(p_j) = \mathbb{E}[y_j|p_j]$ for all $p_j \in \mathcal{P}_j, \forall j \in \{1, \cdots, k\}$. This model $\tilde{M}$ minimizes $R(M)$ and each $R_j(\tilde{M})$. Observe that $R_j(\tilde{M}) = \mathbb{E}_{p_j}\big[\mathsf{Var}[y_j|p_j]\big]$. Therefore, for any minima M^*, $R_j(M^*) = \mathbb{E}_{p_j}\big[\mathsf{Var}[y_j|p_j]\big]$. From equation (4), we obtain that $\mathbb{E}_{p_j}\Big[\big(M^*(p_j) - \mathbb{E}[y_j|p_j]\big)^2\Big] = 0$. From Theorem 1.6.6 in (Ash & Doléans-Dade, 2000), it follows that $M^*(p_j) = \mathbb{E}[y_j|p_j]$ almost everywhere in $\mathcal{P}_j$. □

Theorem. *[Restatement of Theorem 3.2.] If ℓ is the square loss and prompt generation follows equation (3), then the optimal model from equation (2) satisfies,*

$$M^*(p_j) = x_j^\top(\boldsymbol{X}_j^\top\boldsymbol{X}_j + \sigma^2\Sigma^{-1})^{-1}\boldsymbol{X}_j^\top\boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j, \forall j \in \{1, \cdots, k\}$.

Proof. From Lemma 3.1, we know that $M^*(p_j) = \mathbb{E}[y_j|p_j]$ almost everywhere in $\mathcal{P}_j$. We now simplify $\mathbb{E}[y_j|p_j]$ for the data generation provided in equation (3). We follow standard steps of computing the posterior in Bayesian linear regression to obtain the posterior of β conditioned on prefix p_j

$$\begin{aligned}
&\log\big(p(\beta|p_j)\big) = \log p\big(\beta|\boldsymbol{X}_j, \boldsymbol{y}_j, x_j\big) = \log p\big(\beta|\boldsymbol{X}_j, \boldsymbol{y}_j\big) \\
&= \log\big(p\big(\boldsymbol{X}_j, \boldsymbol{y}_j|\beta\big)\big) + \log(p(\beta)) + c \\
&= -\frac{1}{2\sigma^2}\|\boldsymbol{X}_j\beta - \boldsymbol{y}_j\|^2 - \frac{1}{2}\beta^\top\Sigma^{-1}\beta + c \\
&= -\frac{1}{2}(\beta - \mu)^\top\tilde{\Sigma}^{-1}(\beta - \mu) + c
\end{aligned} \tag{5}$$

where $\tilde{\mu} = \tilde{\Sigma}\boldsymbol{X}_j^\top\boldsymbol{y}_j$ and $\tilde{\Sigma} = (\boldsymbol{X}_j^\top\boldsymbol{X}_j + \sigma^2\Sigma^{-1})^{-1}$. Therefore, β conditioned on p_j is a Gaussian distribution with mean $\tilde{\mu}$ and covariance $\tilde{\Sigma}$. Recall

$$y_j = \beta^\top x_j + \varepsilon_j$$

From the linearity of expectation and the expression above for the posterior, it follows

$$\mathbb{E}[y_j|p_j] = \mathbb{E}[y_j|\boldsymbol{X}_j, \boldsymbol{y}_j, x_j] = \mathbb{E}[\beta^\top x_j|\boldsymbol{X}_j, \boldsymbol{y}_j, x_j] = \tilde{\mu}^\top x_j$$

This completes the proof. □

Theorem. *[Restatement of Theorem 3.3] If ℓ is the square loss and prompt generation follows equation (3) with Σ as identity, then in the limit of $\sigma \to 0$ the optimal model from equation (2) satisfies*

$$M^*(p_j) = x_j^\top \boldsymbol{X}_j^+ \boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j, \forall j \in \{1, \cdots, k\}$, where $\boldsymbol{X}_j^+$ is the Moore-Penrose pseudo-inverse of $\boldsymbol{X}_j$.

Proof. For clarity, in this case we make the dependence of $M^*(p_j)$ on σ explicit and instead write it as $M^*(p_j, \sigma)$ We calculate the limit of the ridge regression predictor as σ goes to zero. We obtain

$$\lim_{\sigma \to 0} M^*(p_j, \sigma) = x_j^\top \lim_{\sigma \to 0} (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j = x_j^\top \boldsymbol{X}_j^+ \boldsymbol{y}_j$$

In the simplification above, we used Σ is identity and also used the standard limit definition of Moore-Penrose pseudo-inverse (Albert, 1972).

□

Implications for Theorem 3.3 when Σ is not identity Now consider the more general case when Σ is not identity. In this case, suppose the inverse of $\boldsymbol{X}_j^\top \boldsymbol{X}_j$ exists, which can happen when the rank of $\boldsymbol{X}_j^\top \boldsymbol{X}_j$ is d. In this case, $\lim_{\sigma \to 0} (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top = \boldsymbol{X}_j^+$. To see why this is the case, observe that the map $M^*(p_j, \sigma)$ is well defined for all σ including that at zero and it is also continuous in σ. If the inverse of $\boldsymbol{X}_j^\top \boldsymbol{X}_j$ does not exist, then the limit may not converge to the Moore-Penrose pseudo-inverse. Consider the following example.

Let $\boldsymbol{X}_j = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$ and $\Sigma^{-1} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$, where Σ is invertible and $c \neq 0$.

$$\begin{aligned} (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top &= \frac{1}{c + \sigma^2(ac - b^2)} \begin{bmatrix} c & 0 \\ -b & 0 \end{bmatrix} \\ \lim_{\sigma \to 0} (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top &= \begin{bmatrix} 1 & 0 \\ -\frac{b}{c} & 0 \end{bmatrix} \end{aligned} \tag{6}$$

Thus, $\lim_{\sigma \to 0} (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \neq \boldsymbol{X}_j^+$.

Theorem. *[Restatement of Theorem 3.5] Suppose ℓ is the square loss, β's and x_i's are drawn from an arbitrary distribution with a finite mean and invertible covariance, rest of the prompt generation follows equation (3). In this setting, the solution to equation (2) under Constraint 3.4 satisfies*

$$M^*(p_j) = x_j^\top (\boldsymbol{X}_j^\top \boldsymbol{X}_j + \sigma^2 \Sigma^{-1})^{-1} \boldsymbol{X}_j^\top \boldsymbol{y}_j$$

almost everywhere in $\mathcal{P}_j$, $\forall j \in \{1, \cdots, k\}$.

Proof. Recall that $R(M) = \sum_j R_j(M)$, where $R_j(M) = \mathbb{E}\big[(M(p_j) - y_j)^2\big]$. Let us simplify one of the terms $R_j(M)$.

$$\begin{aligned} R_j(M) &= \mathbb{E}\Big[(M(p_j) - y_j)^2\Big] \\ &= \mathbb{E}\Big[(M(p_j) - y_j)^2\Big] = \mathbb{E}\Big[(M(p_j) - \beta^\top x_j)^2\Big] + \sigma^2 \\ &= \mathbb{E}\Big[\big(m(\boldsymbol{X}_j)\boldsymbol{y}_j - \beta^\top x_j\big)^2\Big] + \sigma^2 \end{aligned} \tag{7}$$

Suppose the covariance of x_j is Λ. We write $\Lambda^{\frac{1}{2}}$ to denote the symmetric positive definite square root of Λ (Such a square root always exists, see Theorem 3 in [2]). We use this to simplify the above expression in equation (7) as follows

$$\begin{aligned} R_j(M) &= \mathbb{E}\left[\left(m(\boldsymbol{X}_j)\boldsymbol{y}_j - \beta^\top x_j\right)^2\right] + \sigma^2 \\ &= \mathbb{E}\left[\left\|\Lambda^{\frac{1}{2}}\left(m(\boldsymbol{X}_j)\boldsymbol{y}_j - \beta\right)\right\|^2\right] + \sigma^2 \\ &= \mathbb{E}\left[\left\|\Lambda^{\frac{1}{2}}\left(m(\boldsymbol{X}_j)\boldsymbol{y}_j\right)\right\|^2\right] + \mathbb{E}\left[\left\|\Lambda^{\frac{1}{2}}\beta\right\|^2\right] - 2\mathbb{E}\left[\boldsymbol{y}_j^\top m(\boldsymbol{X}_j)^\top \Lambda\beta\right] + \sigma^2 \end{aligned} \tag{8}$$

Let us simplfify the first and the third term in the above.

$$\begin{aligned} \mathbb{E}\left[\left\|\Lambda^{\frac{1}{2}}\left(m(\boldsymbol{X}_j)\boldsymbol{y}_j\right)\right\|^2\right] &= \mathbb{E}\left[\boldsymbol{y}_j^\top m(\boldsymbol{X}_j)^\top \Lambda m(\boldsymbol{X}_j)\boldsymbol{y}_j\right] \\ &= \mathbb{E}\left[\beta^\top \boldsymbol{X}_j^\top m(\boldsymbol{X}_j)^\top \Lambda m(\boldsymbol{X}_j)\boldsymbol{X}_j\beta\right] + \sigma^2\mathbb{E}[\mathsf{Trace}[m(\boldsymbol{X}_j)^\top \Lambda m(\boldsymbol{X}_j)]] \end{aligned} \tag{9}$$

In the last simplification above, we use the fact that $\boldsymbol{y}_j = \boldsymbol{X}_j\beta + \boldsymbol{\varepsilon}_j$, where $\boldsymbol{X}_j \in \mathbb{R}^{(j-1)\times d}$ stacks first $j-1$ x_i's and $\boldsymbol{\varepsilon}_j \in \mathbb{R}^{j-1}$ stacks first $j-1$ ε_i's, and that each component of noise is independent and zero mean.

Define $\Theta^1 = \mathbb{E}[\boldsymbol{X}_j^\top m(\boldsymbol{X}_j)^\top \Lambda f(\boldsymbol{X}_j)\boldsymbol{X}_j]$ and $\Theta^2 = m(\boldsymbol{X}_j)^\top \Lambda m(\boldsymbol{X}_j)$. Since $\boldsymbol{X}_j$ is independent of β the above expression simplifies to

$$\mathbb{E}\left[\beta^\top \Theta^1 \beta\right] + \sigma^2\mathsf{Trace}[\Theta^2] = \sum_{i,j} \Theta^1_{i,j}\Sigma_{i,j} + \sigma^2\mathsf{Trace}[\Theta^2] \tag{10}$$

Now let us consider the third term in equation (9).

$$\mathbb{E}\left[\boldsymbol{y}_j^\top m(\boldsymbol{X}_j)^\top \Lambda\beta\right] = \mathbb{E}\left[\beta^\top \boldsymbol{X}_j^\top m(\boldsymbol{X}_j)^\top \Lambda\beta\right] \tag{11}$$

Define $\Gamma = \boldsymbol{X}_j^\top m(\boldsymbol{X}_j)^\top \Lambda$. Since $\boldsymbol{X}_j$ is independent of β the above expression simplifies to

$$\mathbb{E}\left[\beta^\top \Gamma\beta\right] = \sum_{i,j} \Gamma_{i,j}\Sigma_{i,j} \tag{12}$$

From the above simplifications it is clear that the loss depends on prior on β through its mean and covariance only. Therefore, if we use a Gaussian prior with same mean and covariance we obtain the same loss. As a result, we can assume that prior is Gaussian with same mean and covariance and leverage the previous result, i.e., Theorem 3.2. This completes the proof. □

[2] https://www.math.drexel.edu/~foucart/TeachingFiles/F12/M504Lect7.pdf