AN EXPLANATION OF IN-CONTEXT LEARNING AS IMPLICIT BAYESIAN INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large pretrained language models such as GPT-3 have the surprising ability to do in-context learning, where the model learns to do a downstream task simply by conditioning on a prompt consisting of input-output examples. Without being explicitly pretrained to do so, the language model learns from these examples during its forward pass *without parameter updates* on "out-of-distribution" prompts. Thus, it is unclear what mechanism enables in-context learning. In this paper, we study the role of the pretraining distribution on the emergence of in-context learning under a mathematical setting where the pretraining texts have long-range coherence. Here, language model pretraining requires inferring a latent document-level concept from the conditioning text to generate coherent next tokens. At test time, this mechanism enables in-context learning by inferring the shared latent concept between prompt examples and applying it to make a prediction on the test example. Concretely, we prove that in-context learning occurs implicitly via Bayesian inference of the latent concept when the pretraining distribution is a mixture of HMMs. This can occur despite the distribution mismatch between prompts and pretraining data. In contrast to messy large-scale pretraining datasets for in-context learning in natural language, we generate a family of small-scale synthetic datasets (GINC) where Transformer and LSTM language models both exhibit in-context learning. Beyond the theory which focuses on the effect of the pretraining distribution, we empirically find that scaling model size improves in-context accuracy even when the pretraining loss is the same.

1 INTRODUCTION

Large language models (LMs) such as GPT-3 (Brown et al., 2020; Lieber et al., 2021; Wang & Komatsuzaki, 2021; Radford et al., 2019) are pretrained on massive text corpora to simply predict the next word given previous words. They demonstrate the surprising ability to do *in-context learning*, where a LM "learns" to do a downstream task simply by conditioning on a prompt containing input-output pairs. In-context learning in GPT-3 has achieved SOTA results in the LAMBADA (Paperno et al., 2016) completion task and the TriviaQA question answering task (Joshi et al., 2017) (18% and 3% over previous SOTA (Brown et al., 2020)). For example, consider the task of predicting nationalities from names. A prompt (Figure 1) is constructed by concatenating independent "training" examples (e.g., "Albert Einstein was German") followed by a "test example" ("Marie Curie was"). The LM conditions on this prompt, presented as a contiguous sequence of examples, to make a prediction. In this example, GPT-3 places the largest probability on the correct output

$$p(\text{“Polish”} \mid \text{“Albert Einstein was German \textbackslash n Mahatma Gandhi was Indian \textbackslash n Marie Curie was”})$$

seemingly by inferring the task (names to nationalities) from the examples. Intuigingly, GPT-3 was not explicitly pretrained to learn from examples and does not optimize any parameters on the downstream task. Further, the distribution of prompts are quite different from real text – because the examples are independently generated, the transitions between examples can be very unnatural. How does in-context learning work? Our understanding is limited since (i) real pretraining data is messy and (ii) in-context learning has so far required large-scale datasets and models.

In this paper, we introduce a simple pretraining distribution which gives rise to in-context learning. To generate a document, we first draw a latent concept θ, which parameterizes the transitions of a Hidden Markov Model (HMM) (Baum & Petrie, 1966), then sample a sequence of observations (tokens) from the HMM (see Figure 2). This latent variable structure is common in topic models such as LDA (Blei et al., 2003; Gruber et al., 2007). To capture long-range coherence in the pretraining data, an LM must

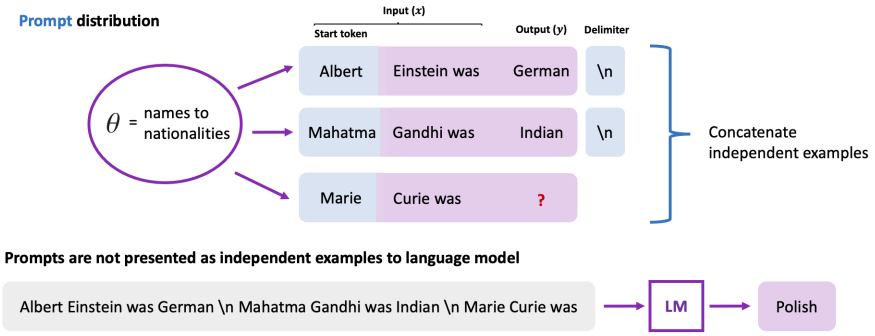

Figure 1: **(Top)** In the in-context prompt distribution, we first sample a latent prompt concept, which encodes relationships (e.g., names to nationalities) that occur within a document, and then generate independent examples conditioned on this shared prompt concept. Each example samples a start token (e.g., Albert) from the prompt start distribution (e.g., a distribution over names) then an input x (e.g., Albert Einstein was) and the output token y (e.g., German) from the pretraining distribution (purple) conditioned on the start token and the prompt concept. A fixed delimiter (e.g., newline) separates the examples. **(Bottom)** The in-context prompt is presented as a contiguous sequence rather than independent examples and are thus "out-of-distribution" to the LM. The model must infer the shared prompt concept to generate the correct answer to "Marie Curie was" (e.g., "Polish", not "a scientist")

infer the latent concept across multiple sentences to generate coherent continuations. When conditioning on a prompt, in-context learning occurs when the LM also infers a shared latent concept across examples to make a prediction. We assume the LM fits the pretraining distribution exactly with enough data and expressivity, so that the question of in-context learning becomes characterizing the conditional distribution of completions given prompts $p(\text{output}|\text{prompt})$ under the pretraining distribution, where the prompt is generated from a prompt distribution p_{prompt}. The pretraining distribution marginalizes out the latent concepts (and HMM hidden states), so this conditional distribution is the *posterior predictive distribution*. Thus, in-context learning can be viewed as implicit Bayesian inference.

Ideally, the posterior predictive distribution should concentrate on the correct output as the number of in-context examples increases, but the main challenge is that prompts are not presented as a set of independent examples but as a concatenated sequence (Figure 1). Thus, the prompts are from a different distribution than the pretraining distribution. The canonical asymptotic tool in Bayesian methods is the Bernstein-von Mises theorem (van der Vaart, 1998; Kleijn & van der Vaart, 2012; Gunst & Shcherbakova, 2008), which typically assumes observations are independent and/or drawn from the same distribution as the model, both of which are not satisfied. However, we prove that in-context learning can emerge in the limit of infinite examples when the signal about the latent concept in each prompt example is larger than the error stemming from the mismatch between the pretraining and prompt distributions. Additionally, we prove that the in-context learning error can be bounded by a function of the length of each example—longer example sequences provide more signal on the latent concept.

As a companion to this theory, we created the **G**enerative **IN**-**C**ontext learning dataset (GINC), which is a small-scale synthetic dataset which we can use to study in-context learning without relying on large-scale, messy real-world data. We find that both Transformer (Vaswani et al., 2017) and LSTM (Hochreiter & Schmidhuber, 1997) language models trained on such datasets exhibit in-context learning. We verify intuitions from the theory, showing that the accuracy of in-context learning improves with the number of examples and example length. Ablations of the GINC dataset show that the mixture-over-concepts structure in the pretraining distribution is crucial to the emergence of in-context learning. The experiments also bring up open questions which go beyond our theory, which only studies the pretraining distribution. We find that scaling up the number of model parameters steadily improves the in-context accuracy despite achieving the same pretraining loss, showing that larger models may improve in-context learning beyond increasing the capacity for memorizing the training data better.

2 In-context learning setting

Pretraining distribution. In our framework, a latent concept θ from a family of concepts Θ defines a distribution over observed tokens o from a vocabulary $\mathcal{O}$. To generate a document, we first sample

a concept from a prior over concepts $p(\theta)$ and then sample a sequence given the concept. We model the pretraining distribution as a mixture of token distributions described by latent concepts. Each pretraining document is a length T sequence:

$$p(o_1,...,o_T) = \int_{\theta \in \Theta} p(o_1,...,o_T|\theta)p(\theta)d\theta \tag{1}$$

In our theoretical analysis, we assume $p(o_1,...,o_T|\theta)$ is described by the distribution over observations from a Hidden Markov Model (HMM). The concept θ determines the transition probability matrix of the HMM hidden states $h_1,...,h_T$ from a hidden state set $\mathcal{H}$.

Prompt distribution. The prompt distribution p_{prompt} generates prompts for in-context learning. Each prompt is generated with respect to a prompt concept θ^*, which is shared by all the prompt examples.

A prompt example is composed of an input token sequence x (e.g., Albert Einstein was) followed by an output token y (e.g., German). The prompt is a concatenation of n independent training examples and 1 test input x_{test}, where the goal is the predict the test output y_{test} by predicting the next token. In particular, the i-th training example O_i consists of an input $x_i = O_i[1:\ k-1]$ (the first $k-1$ tokens of the example) followed by an output token $y_i = O_i[k]$ at the end[1]. The i-th training example is independently generated as follows:

1. Generate a start hidden state h_i^{start} from a *prompt start distribution* π_{prompt}.
2. Given h_i^{start}, generate the example sequence $O_i = [x_i, y_i]$ from $p(O_i|h_i^{\text{start}}, \theta^*)$, the *pretraining distribution* conditioned on a prompt concept θ^*.

The test example $x_{\text{test}} = x_{n+1}$ is sampled similarly. The resulting prompt sequence concatenates the training examples (denoted S_n) and test example (x_{test}) separated by a special delimiter token o^{delim}:

$$[S_n, x_{\text{test}}] = [x_1, y_1, o^{\text{delim}}, x_2, y_2, o^{\text{delim}}, ..., x_n, y_n, o^{\text{delim}}, x_{\text{test}}] \sim p_{\text{prompt}}. \tag{2}$$

Prompts are out-of-distribution. We provide a simple illustration for why prompts are OOD. Suppose a pretraining document is `a b c d...x y z a b c d...y z...` where the latent concept enforces alphabetical order with high probability. A prompt based on this concept might be `a b \n x y \n f g \n c`, with independent examples and a test input `c`. The transitions between examples are unnatural (e.g., `x` rarely follows closely after `b`). Thus, prompts are low probability (OOD) sequences under the pretraining distribution. Even though the examples were independently generated, we can no longer factorize the examples under the pretraining distribution.

In-context predictor and task. For in-context learning, the output target y for each example x is sampled according to $p_{\text{prompt}}(y|x)$:

$$y_{\text{test}} \sim p_{\text{prompt}}(y|x_{\text{test}}) = \mathbb{E}_{h_{\text{test}}^{\text{start}} \sim \pi_{\text{prompt}}(h_{\text{test}}^{\text{start}}|x_{\text{test}})} \left[p(y|x_{\text{test}}, h_{\text{test}}^{\text{start}}, \theta^*) \right]. \tag{3}$$

where $h_{\text{test}}^{\text{start}}$ denotes the hidden state corresponding to the first token of x_{test} and $\pi_{\text{prompt}}(h_{\text{test}}^{\text{start}}|x_{\text{test}})$ is defined as $p_{\text{prompt}}(x_{\text{test}}|h_{\text{test}}^{\text{start}})\pi_{\text{prompt}}(h_{\text{test}}^{\text{start}})/p_{\text{prompt}}(x_{\text{test}})$. The target distribution differs from the pretraining distribution p on the distribution of $h_{\text{test}}^{\text{start}}$.

The central question is, when the number of examples grows, does the posterior predictive distribution of the pretraining distribution given the prompt concentrate on the correct output? Towards this, we study the in-context predictor $f_n(x_{\text{test}}) = \text{argmax}_y p(y|S_n, x_{\text{test}})$ and its expected 0-1 error:

$$L_{\text{0-1}}(f_n) = \mathbb{E}_{x_{\text{test}}, y_{\text{test}} \sim p_{\text{prompt}}}[\mathbf{1}[f_n(x_{\text{test}}) \neq y_{\text{test}}]]. \tag{4}$$

2.1 Assumptions

We detail the assumptions in our framework, including the structure of delimiters and regularity assumptions. We first assume that there exists a subset of *delimiter hidden states* $\mathcal{D}$ which generates the special delimiter token o^{delim} deterministically.

Assumption 1 (Delimiter hidden states)**.** *Let the delimiter hidden states $\mathcal{D}$ be a subset of $\mathcal{H}$. For any $h \in \mathcal{D}$ and θ, $p(o^{delim}|h,\theta) = 1$ and for any $h' \notin \mathcal{D}$, $p(o^{delim}|h,\theta) = 0$.*

[1] Example length k is fixed for simplicity, but in practice, it will vary — our analysis can be extended to variable k, but we leave this to future work.

Thus if we observe the special delimiter token o^{delim}, then the corresponding hidden state must be in $\mathcal{D}$. However, it does not reveal the particular hidden state in $\mathcal{D}$ nor the latent prompt concept θ^*.

The delimiter token is usually a token that can appear in a broad range of contexts (e..g, newline). The delimiter ideally does not distract from the examples — for example, an adversarial delimiter could look like part of the input x. To mitigate these scenarios, we assume that no delimiter (e.g., newline) is significantly more likely under one concept rather than another.

Assumption 2 (Bound on delimiter transitions)**.** *For any delimiter state $h^{delim} \in \mathcal{D}$ and any hidden state $h \in \mathcal{H}$, the probability of transitioning to a delimiter hidden state under θ is upper bounded $p(h^{delim}|h,\theta) < c_1$ for any $\theta \in \Theta \setminus \{\theta^*\}$, and is lower bounded $p(h^{delim}|h,\theta^*) > c_2 > 0$ for θ^*. Additionally, the start hidden state distribution over delimiters is uniform for all concepts θ: $p(h^{delim}|\theta) = \frac{1}{|\mathcal{D}|}$ for all $\theta \in \Theta$ and $h^{delim} \in \mathcal{D}$.*

We also assume the prompt concept θ^* is in the family Θ, which is a broad set of concepts.

Assumption 3 (Well-specification)**.** *The prompt concept θ^* is in Θ.*

Even though the pretraining distribution is broad, the prompt is still out-of-distribution for the pretraining distribution since it concatenates independent examples. Finally, if the prompt has zero probability under the prompt concept θ^*, then Bayesian inference will not be able to infer the prompt concept as in Section 3.1. The following are regularity assumptions which mainly ensure that the prompt is not zero probability under θ^*.

Assumption 4 (Regularity)**.** *The pretraining distribution p satisfies: 1) Lower bound on transition probability for the prompt concept θ^*: for any pair of hidden states $h,h' \in \mathcal{H}$, $p(h|h',\theta^*) > c_3 > 0$. 2) All tokens can be emitted: for every symbol o, there is some hidden state $h \in \mathcal{H}$ such that $p(o|h,\theta^*) > c_4 > 0$. 3) The hidden state Markov chain for concept θ has stationary distribution π^θ over $\mathcal{H}$. 4) The prior $p(\theta)$ has support over the entire concept family Θ and is bounded above everywhere.*

We denote the stationary distribution conditioned on x to be $\pi^\theta(h|x) = (p(x|h,\theta)\pi^\theta(h))/p(x|\theta)$.

3 Theoretical analysis

We prove that in the limit of infinite examples, conditioning the pretraining distribution on the prompt results in in-context learning provided that a *distinguishability* condition holds — the prompt concept θ^* is distinct enough from the other concepts in Θ (e.g., when Θ is a discrete set). In this case, the expected error is optimal. When distinguishability does not hold (e.g, Θ is continuous-valued), we show that the expected error still decreases with example length.

3.1 High-level approach

In-context learning in GPT-3 is often done with greedy decoding or nucleus sampling (Holtzman et al., 2020), which samples high probability completions. Following this practice, we analyze the most likely prediction over the *pretraining* distribution conditioned on the prompt, which is sampled from the *prompt* distribution. Our goal is to show that $\text{argmax}_y p(y|S_n,x_{\text{test}}) \to \text{argmax}_y p_{\text{prompt}}(y|x_{\text{test}})$ as the number of examples n goes to infinity.

In the following, assume that the prompt has non-zero probability under the pretraining distribution p given θ^*, meaning that $p(S_n,x_{\text{test}}|\theta^*) > 0$. We expand $p(y|S_n,x_{\text{test}})$ to analyze its limit:

$$
\begin{aligned}
p(y|S_n,x_{\text{test}}) &= \int_\theta p(y|S_n,x_{\text{test}},\theta)p(\theta|S_n,x_{\text{test}})d\theta \\
&\propto \int_\theta p(y|S_n,x_{\text{test}},\theta)p(S_n,x_{\text{test}}|\theta)p(\theta)d\theta \quad \text{(Bayes' rule, drop the constant } \frac{1}{p(S_n,x_{\text{test}})}\text{)} \\
&= \int_\theta \sum_{h^{\text{start}}_{\text{test}} \in \mathcal{H}} p(y|x_{\text{test}},h^{\text{start}}_{\text{test}},\theta)p(h^{\text{start}}_{\text{test}}|S_n,x_{\text{test}},\theta)\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}p(\theta)d\theta \qquad (5) \\
&\qquad \text{(Law of total prob, Markov property, divide by } p(S_n,x_{\text{test}}|\theta^*) \text{ (a constant))} \\
&= \int_\theta \sum_{h^{\text{start}}_{\text{test}} \in \mathcal{H}} p(y|x_{\text{test}},h^{\text{start}}_{\text{test}},\theta)p(h^{\text{start}}_{\text{test}}|S_n,x_{\text{test}},\theta)\exp(n \cdot r_n(\theta))p(\theta)d\theta \qquad (6)
\end{aligned}
$$

where $r_n(\theta) = \frac{1}{n}\log\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}$. In Theorem 1, we prove that under a distinguishability condition, $\exp(n \cdot r_n(\theta)) \to 0$ for all concepts θ except the prompt concept θ^*, where $\exp(n \cdot r_n(\theta^*)) = 1$. The

only nonzero term in the integral is when $\theta = \theta^*$, and thus the prompt concept is "selected" as a consequence of Bayesian inference. Note that we can exchange limits and integrals here since the probabilities involved in the first steps are bounded (dominated convergence). In Lemma 3, we show that the distribution of the hidden start state for the test example $p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta^*)$ converges to the stationary distribution $\pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}})$ conditioned on x_{test}. Putting these together with Equation 6, we have that the in-context predictor infers prompt concept θ^*:

$$\operatorname*{argmax}_y p(y|S_n,x_{\text{test}}) \to \operatorname*{argmax}_y \mathbb{E}_{h_{\text{test}}^{\text{start}} \sim \pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}})}[p(y|x_{\text{test}},h_{\text{test}}^{\text{start}},\theta^*)] \tag{7}$$

Recall that the test label is sampled from $p_{\text{prompt}}(y|x_{\text{test}}) = \mathbb{E}_{h_{\text{test}}^{\text{start}} \sim \pi_{\text{prompt}}(h_{\text{test}}^{\text{start}}|x_{\text{test}})}[p(y|x_{\text{test}},h_{\text{test}}^{\text{start}},\theta^*)]$. If the prompt start distribution π_{prompt} equals the stationary distribution π^{θ^*}, then the conditional state distributions $\pi_{\text{prompt}}(h_{\text{test}}^{\text{start}}|x_{\text{test}}) = \pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}})$ are the same between pretraining and prompting. Thus, the in-context predictor makes the optimal prediction when $\pi_{\text{prompt}} = \pi^{\theta^*}$.

3.2 Heuristic derivation

Recall from Section 3.1 that if $\exp(n \cdot r_n(\theta)) \to 0$ for all $\theta \neq \theta^*$, then Bayesian inference "selects" the prompt concept through marginalization. To do this, we focus on showing that $r_n(\theta)$, the average log-likelihood ratio between θ and θ^*, converges to a negative constant, and thus nr_n goes to $-\infty$.

The main technical challenge is to handle the sequence-of-examples structure of the prompt, which makes all the examples dependent with respect to the pretraining distribution. Our approach uses properties of delimiter tokens to approximately factorize the examples, with constant error per example. We let $O_i^{\text{ex}} = [o_{i-1}^{\text{delim}}, O_i]$ be the i-th input-output pair and the previous delimiter together for $i > 1$ and define $O_1^{\text{ex}} = O_1$. Expanding the likelihood term inside $r_n(\theta)$, our goal is to show

$$p(S_n,x_{\text{test}}|\theta) = p(x_{\text{test}}|S_n,\theta)p(S_n|\theta) \approx \prod_{i=1}^{n} O(1)p(O_i|\theta) \tag{8}$$

To show this, we expand $p(S_n|\theta)$ with the chain rule, and with Assumption 4 (to bound $p(x_{\text{test}}|S_n,\theta)$ by $O(1)$) it can be shown that

$$p(x_{\text{test}}|S_n,\theta)p(S_n|\theta) \approx \prod_{i=1}^{n} O(1)p(O_i^{\text{ex}}|O_{1:i-1}^{\text{ex}},\theta). \tag{9}$$

We then marginalize $p(O_i^{\text{ex}}|O_{1:i-1}^{\text{ex}},\theta)$ over the hidden state h_{i-1}^{delim} corresponding to the delimiter in $O_i^{\text{ex}} = [o_{i-1}^{\text{delim}}, O_i]$:

$$\prod_{i=1}^{n} O(1)p(O_i^{\text{ex}}|O_{1:i-1}^{\text{ex}},\theta) = \prod_{i=1}^{n} O(1) \sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta) \approx \prod_{i=1}^{n} O(1)p(O_i|\theta) \tag{10}$$

While summing over $\mathcal{H}$ would be a trivial equality, we can replace $\mathcal{H}$ with the set of delimiter hidden states $\mathcal{D}$ since $p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta) = 0$ for non-delimiter hidden states (Assumption 1). We used in the first equality that $O_{1:i-1}^{\text{ex}} \to h_{i-1}^{\text{delim}} \to O_i^{\text{ex}}$ forms a Markov chain and $p(o_{i-1}^{\text{delim}}|h_{i-1}^{\text{delim}}) = 1$ (Assumption 1) to change O_i^{ex} to O_i. Finally, we can show using properties of delimiter hidden states (Assumption 2) that $p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta) \approx O(1)p(h_{i-1}^{\text{delim}}|\theta)$, which allows us to marginalize out h_{i-1}^{delim} in the second step. Therefore, we can upper bound $r_n(\theta)$ as

$$r_n(\theta) \leq \frac{1}{n}\left(O(n) + \sum_{i=1}^{n} \log \frac{p(O_i|\theta)}{p(O_i|\theta^*)}\right) \to O(1) + \mathbb{E}_{O \sim p_{\text{prompt}}}\left[\log \frac{p(O|\theta)}{p(O|\theta^*)}\right]. \tag{11}$$

Since all tokens after the start token in an example O are drawn from the pretraining distribution p conditioned on the prompt concept θ^*, the expectation in Equation 11 becomes a negative KL divergence for tokens 2 through k. There are $O(k)$ KL terms and only $O(1)$ other error terms, which come from the distribution mismatch between the prompt and pretraining distributions. If the KL terms are larger than the error terms, then $r_n(\theta)$ has a negative limit. If this holds for all $\theta \neq \theta^*$, then we have $\exp(n \cdot r_n(\theta)) \to 0$ for all $\theta \neq \theta^*$, enabling in-context learning.

3.3 FORMAL RESULTS

3.3.1 IN-CONTEXT LEARNING UNDER DISTINGUISHABILITY

We define a distinguishability condition which formalizes when in-context learning occurs. Letting p^j_θ be the output distribution of the j-th token given the previous tokens, the condition depends on KL divergence between $p^j_{\theta^*}$ and p^j_θ as well as error terms $\epsilon^\theta_{\text{start}}$ and $\epsilon^\theta_{\text{delim}}$ coming from the distribution mismatch between the prompt and pretraining distributions at the start and delimiter token for each example:

$$p^j_\theta(o) = p(O[j]=o \,|\, O[1:\, j-1],\theta), \quad KL_j(\theta^* \| \theta) = \mathbb{E}_{O[1:\, j-1]\sim p_{\text{prompt}}}[KL(p^j_{\theta^*} \| p^j_\theta)] \tag{12}$$

$$\epsilon^\theta_{\text{start}} = \mathbb{E}_{O[1]\sim p_{\text{prompt}}}\left[\log \frac{p(O[1]|\theta)}{p(O[1]|\theta^*)}\right], \quad \epsilon^\theta_{\text{delim}} = (\log(c_1) - \log(c_2)). \tag{13}$$

Condition 1 (Distinguishability)**.** *We define θ^* to be distinguishable if for all $\theta \in \Theta, \theta \neq \theta^*$,*

$$\sum_{j=2}^{k} KL_j(\theta^* \| \theta) > \epsilon^\theta_{start} + \epsilon^\theta_{delim}. \tag{14}$$

This condition operates on the level of one example. When the signal from KL divergence (LHS) is larger than the error terms, Equation 14 is satisfied. For longer example lengths k, the LHS increases and the condition is more likely satisfied. Intuitively, longer example lengths increases the proportion of the prompt sampled from the pretraining distribution. Under Condition 1, we show that the in-context predictor asymptotically achieves the optimal expected error.

Theorem 1. *Assume the assumptions in Section 2.1 hold. If Condition 1 holds, then as $n \to \infty$ the prediction according to the pretraining distribution is*

$$\operatorname*{argmax}_y p(y|S_n, x_{test}) \to \operatorname*{argmax}_y \mathbb{E}_{h^{start}_{test} \sim \pi^{\theta^*}(h^{start}_{test}|x_{test})}[p(y|x_{test}, h^{start}_{test}, \theta^*)]. \tag{15}$$

As a corollary, if the prompt start distribution π_{prompt} is the stationary distribution π^{θ^*} of the hidden state chain for θ^*, then the in-context predictor f_n achieves the optimal expected 0-1 error:

$$\lim_{n\to\infty} L_{0\text{-}1}(f_n) = \inf_f L_{0\text{-}1}(f). \tag{16}$$

When the prompt start distribution π_{prompt} matches the stationary distribution π^{θ^*}, the in-context predictor predicts the most likely token from the same predictive distribution as the test example (Equation 3). When this is not true, there may be additional distribution shift.

3.3.2 NON-DISTINGUISHABLE CASE

Even when the distinguishability condition does not hold, can we give an error guarantee? Note that Condition 1 fails when there is some $\theta \neq \theta^*$ for which the KL between θ and θ^* is less than the error terms. However, this means that the output distributions of θ and θ^* are close in KL. We leverage this to prove that the expected 0-1 error decreases with the example length k under two different settings.

Continuity. Our first result relies on a continuity assumption between the concept parameter and its corresponding output distribution. Our assumption is based on prior works (Kleijn & van der Vaart, 2012), where the KL divergence is assumed to have a 2nd-order Taylor expansion.

Theorem 2. *Let the set of θ which does not satisfy Equation 14 in Condition 1 to be $\mathcal{B}$ and assume $\pi_{prompt} = \pi^{\theta^*}$. Assume that KL divergences have a 2nd-order Taylor expansion[2] around θ^*:*

$$\forall j,\ KL_j(\theta^* \| \theta) = \frac{1}{2}(\theta - \theta^*)^\top I_{j,\theta^*}(\theta - \theta^*) + O(\|\theta - \theta^*\|^3) \tag{17}$$

where I_{j,θ^*} is the Fisher information matrix of the j-th token distribution with respect to θ^*. Let $\gamma_{\theta^*} = \frac{\max_j \lambda_{max}(I_{j,\theta^*})}{\min j \lambda_{min}(I_{j,\theta^*})}$ where $\lambda_{max}, \lambda_{min}$ return the largest and smallest eigenvalues. Then as $n \to \infty$, the 0-1 risk of the in-context learning predictor f_n is bounded as

$$\lim_{n\to\infty} L_{0\text{-}1}(f_n) \leq \inf_f L_{0\text{-}1}(f) + g^{-1}\left(O\left(\frac{\gamma_{\theta^*} \sup_{\theta\in\mathcal{B}}(\epsilon^\theta_{start} + \epsilon^\theta_{delim})}{k-1}\right)\right) \tag{18}$$

where $g(\delta) = \frac{1}{2}((1-\delta)\log(1-\delta) + (1+\delta)\log(1+\delta))$ is a calibration function (Steinwart, 2007; Ávila Pires & Szepesvári, 2016) for the multiclass logistic loss for $\delta \in [0,1)$.

[2] This requires the KL divergence to be twice differentiable.

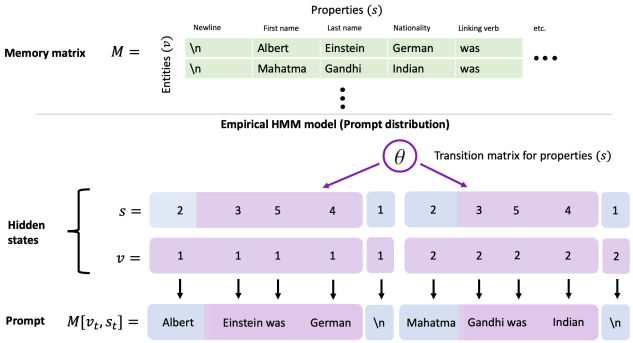

Figure 2: The GINC dataset generates sequences from a mixture of HMMs. The HMM hidden states consist of entities (v) and properties (s), which index into a memory matrix to produce the observed token. The entity and property sequences are sampled from independent Markov chains. The concept parameter θ is the transition matrix for properties, which defines relations between properties. In this example, the sequence of properties [2,3,5,4] relates names to nationalities, defining the in-context task.

Since the inverse calibration function g^{-1} is roughly linear in ϵ for $\epsilon \leq 0.7$, the excess risk roughly decreases as $O(1/k)$. When the "worst-case condition number" γ_{θ^*} of the Fisher information matrices is smaller (well-conditioned), the error decreases. Intuitively, this means that there is no direction to vary θ^* in which the output distribution will sharply change. As a consequence, the concepts θ that are not distinguishable (via Equation 14) from the prompt concept θ^* parameterize distributions that produce similar outputs to the prompt concept and thus achieve a small error.

Varying-length test examples. In the setting where the length of x_{test} is random (uniformly from 2 to k), we can give a similar error guarantee without the continuity assumption from Theorem 2.

Theorem 3. *Let the set of θ which does not satisfy Equation 14 in Condition 1 to be $\mathcal{B}$ and assume $\pi_{prompt} = \pi^{\theta^*}$. Let the length of the test example x_{test} be uniformly distributed between 2 and k, for $k \geq 2$. As $n \to \infty$, the 0-1 risk of the in-context learning predictor f_n is bounded as*

$$\lim_{n\to\infty} L_{0\text{-}1}(f_n) \leq \inf_f L_{0\text{-}1}(f) + g^{-1}\left(\frac{\sup_{\theta\in\mathcal{B}}(\epsilon^{\theta}_{start} + \epsilon^{\theta}_{delim})}{k-1}\right). \tag{19}$$

Instead of measuring only the error at the k-th token, we average the prediction error on the 2nd to k-th tokens. However, we leave bridging the mismatch between training examples, which are consistently length k, and test examples, which have random length, to future work.

4 Simulations

We generate the GINC dataset from a mixture of HMMs and show that Transformers (Vaswani et al., 2017) and LSTMs (Hochreiter & Schmidhuber, 1997) trained on GINC exhibit in-context learning.

4.1 GINC Pretraining Dataset

We consider a pretraining distribution that is a mixture of HMMs with an interpretable hidden state structure and emission distribution (see Appendix E for details and a concrete example). The HMM hidden state $h_t = [s_t, v_t]$ at time t is composed of an *entity* $v_t \in \{1,...,|\mathcal{V}|\}$ (e.g., Einstein) and a *property* $s_t \in \{1,...,|\mathcal{S}|\}$ (e.g., nationality, first name, last name, other grammatical tokens). We model the entities and properties as independent Markov chains (i.e., a factorial HMM (Ghahramani & Jordan, 1997)), while the emissions depend on both. In pretraining documents, we expect that the entities (e.g., Einstein) change slowly over time while and their properties (e.g., nationality) change quickly to generate natural sentences. We implement this by ensuring that the probability of transitioning to the same entity index in the next step is at least 0.9. The emission distribution depends on a memory matrix M with $|\mathcal{V}|$ rows and $|\mathcal{S}|$ columns (Figure 2). At step t, we use the entity v_t and property s_t to index into the memory matrix. In particular, the observed tokens are deterministic with $p(o_t|h_t) = 1$ if $o_t = M[v_t, s_t]$.

Concept parameter. The concept parameter is the property transition matrix, which encodes many relations between properties to generate a diverse pretraining document. The entity transition matrix is the same for all concepts. During in-context learning, the prompt start distribution and the concept

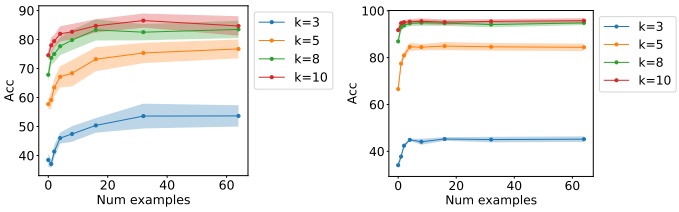

Figure 3: In-context accuracy (95% intervals) of Transformers (left) and LSTMs (right) on the GINC dataset. Accuracy increases with number of examples n and length of each example k.

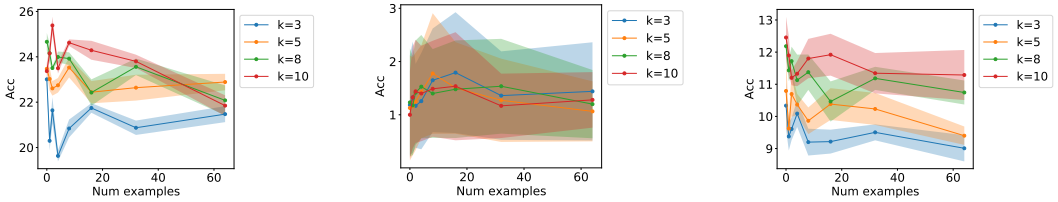

Figure 4: Ablation studies for 4 layer Transformers on the GINC dataset with vocab size 50. **(Left)** When pretrained with only one concept, in-context learning fails. **(Middle)** When the pretraining data has random transitions, the model sees all token transitions but in-context learning fails. **(Right)** When prompt concepts are out-of-distribution (and the model is pretrained normally), in-context learning fails.

together determine the in-context task. While the prompt concept θ^* may have a high probability transition between names and nationalities, another concept may tend to transition to birthplaces after names. Here, if the prompt start distribution focuses on names (with random entities for each example), then the prompt examples (generated according to θ^*) will largely focus on the transition between names and nationalities, visiting part of the total hidden state space. In our experiments, we define a uniform mixture of HMMs over 5 concepts to generate 1000 documents with $\tilde{1}0$ million tokens total.

4.2 GINC PROMPT DISTRIBUTION

We generate prompts according to the theoretical setup, where we first sample a concept θ uniformly at random, then use it to generate all the prompt examples. The prompt start distribution is chosen to be uniform over entities but with a fixed starting property that is chosen randomly for each prompt, for consistency in the task. We generate prompts with 0 to 64 training examples and example lengths $k = \{3,5,8,10\}$ (2500 prompts for each setting). The target token y_{test} is taken to be the most likely output $\text{argmax}_y p_{\text{prompt}}(y|x_{\text{test}})$ instead of sampling to reduce the intrinsic error.

4.3 SIMULATION RESULTS

We train GPT-2-based Transformer (Radford et al., 2019) and LSTM language models on three versions of the GINC dataset with vocabulary sizes 50, 100, and 150, then evaluate the in-context accuracy (see Appendix E.2, E.3 for model details). We average all results over 5 pretraining runs. Figure 3 shows that for both Transformer and LSTMs, in-context accuracy improves as the number of prompt examples n and the example length k increase, verifying our theory.

Ablations and out-of-distribution concepts. We ablate the role of the mixture-of-concepts structure in GINC. In Figure 4 (left), we pretrain a 4 layer Transformer on data with only one concept (removing the prior) from Θ, resulting in in-context accuracy curves that do not increase with the number of examples. Figure 4 (middle) shows that when pretraining on random data, which contains all possible transitions between tokens, in-context learning also fails. Therefore, the mixture-of-concepts structure is important to in-context learning and simply seeing a diverse set of token transitions does not enable in-context learning. Finally, we test a 4 layer Transformer pretrained on GINC on randomly generated concepts that are not in the pretraining family of concepts. We generate 5 additional concepts and use these to generate prompts. Figure 4 (right) shows that in-context learning also fails for these novel concepts.

Effect of model size and architecture. Figure 5 shows that increasing the Transformer's size (4, 12, 16 layers) increases in-context accuracy, as in Brown et al. (2020). Table 6 shows that even though larger Transformers may have the same pretraining loss (e.g., 12 and 16 layer Transformers both get 1.33 validation loss for vocab size 50), the in-context accuracy still improves (81% to 85% from 12 to 16

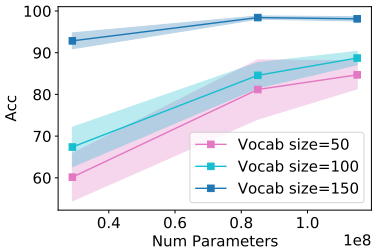

Figure 5: In-context accuracy (95% intervals) of Transformers improves with model size on GINC for vocabulary sizes 50, 100, and 150.

Model	# Params	Train loss (pretraining)	Val loss (pretraining)	In-context Acc
Vocab size 50, $k=10, n=64$				
Transformer (4 layer)	29M	1.49	1.50	60.2 ± 5.7
Transformer (12 layer)	85M	1.31	1.33	81.2 ± 7.1
Transformer (16 layer)	115M	1.31	1.33	84.7 ± 3.4
LSTM	28M	1.31	1.35	95.8 ± 1.11
Vocab size 100, $k=10, n=64$				
Transformer (4 layer)	29M	1.58	1.59	67.4 ± 4.7
Transformer (12 layer)	85M	1.40	1.42	84.6 ± 3.0
Transformer (16 layer)	115M	1.41	1.43	88.7 ± 1.6
LSTM	28M	1.43	1.44	95.8 ± 1.54
Vocab size 150, $k=10, n=64$				
Transformer (4 layer)	29M	1.44	1.45	92.8 ± 1.9
Transformer (12 layer)	85M	1.27	1.28	98.4 ± 0.4
Transformer (16 layer)	115M	1.27	1.28	98.1 ± 0.5
LSTM	28M	1.26	1.31	99.2 ± 1.06

Figure 6: In-context accuracies (95% intervals) on GINC with vocab sizes (50, 100, 150) for Transformers and LSTMs. Accuracy improves with scale even though the pretraining loss may be the same.

layers), suggesting that larger models can improve in-context learning beyond increasing memorization capacity. Finally, the architecture also plays a role — LSTMs consistently outperform Transformers on GINC despite having fewer parameters, perhaps due to the similarity between HMMs and LSTMs. We leave analysis of the effect of model architecture and model scaling as an open question.

5 Discussion and Related Work

Learning via Bayesian inference and misspecification. The Bernstein-von Mises theorem states that under some conditions, the posterior distribution of a parameter in a Bayesian parametric model is asymptotically normal and centered on the MLE (van der Vaart, 1998). However, this does not apply for in-context learning since the prompt examples are not presented as independent samples but as one contiguous sequence. We also focus on the posterior predictive distribution rather than the parameter. Gunst & Shcherbakova (2008) show a Bernstein-von Mises-type result for observations from an HMM, but do not handle observations from a different distribution. However, we hope to show more precise asymptotic results in the future about the posterior distribution and give results under misspecification (Kleijn & van der Vaart, 2012).

Topic models. Topic models such as LDA (Blei et al., 2003) also use mixture models of text, but learning is typically done through algorithms such as Expectation-Maximization (Dempster et al., 1977), variational inference (Jordan et al., 1999), or MCMC (Metropolis et al., 1953; Hastings, 1970). Since in-context learning occurs as a byproduct of a forward pass, we focus on learning as a natural result of marginalization in Bayesian inference without an explicit inference algorithm.

Modeling the pretraining distribution with HMMs. Wei et al. (2021a) also use an HMM in their generative model of the pre-training distribution. However, their goal is different – they analyze how pre-trained representations learned with masked language modeling (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2020; Clark et al., 2020) can be adapted for downstream tasks through linear probing and prompt tuning (Li & Liang, 2021; Lester et al., 2021) rather than in-context learning.

Bridging the mismatch between pretraining and prompting. Prior work shows that finetuning LMs on text with a prompting format improves its zero-shot performance (Wei et al., 2021b; Sanh et al., 2021). Relatedly, Jiang et al. (2020); Schick & Schütze (2021); Shin et al. (2020); Gao et al. (2021) optimize prompt templates for better few-shot finetuning. Zhao et al. (2021); Holtzman et al. (2021) improve in-context accuracy via calibration or renormalization, a form of adaptation to the prompt distribution. These methods support our theoretical intuitions that reducing the prompt distribution mismatch would improve in-context learning.

Meta-learning. Meta-learning methods that train a sequence model to learn from examples (Ravi & Larochelle, 2017) also performs learning during the forward pass of a model. However, meta-learning models are trained to learn, while in-context learning emerges from LM pretraining.

6 Conclusion

We cast in-context learning as implicit Bayesian inference, where the pretrained LM implicitly infers a concept when making a prediction. We show that in-context learning occurs when the pre-training distribution is a mixture of HMMs. Our work provides a first step towards understanding in-context learning, which we hope will provide insight for improving pretraining and prompting.

7 ETHICS

Since the datasets in this paper are synthetically generated, there are no privacy/security issues to report. We investigate the emergent behaviors in large models, which will ultimately require careful consideration of environmental factors (energy usage), bias stemming from pretraining data, and others.

8 REPRODUCIBILITY

The source code for the experiments are included as a supplementary material. Proofs for theoretical results are included in the appendix, and assumptions are in Section 2.1. The datasets used in the experiments can be generated from the supplementary code. The experimental results are averaged over 5 seeds and 95% intervals are reported.

REFERENCES

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

D. Blei, Andrew Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, 3:993–1022, 2003.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. Electra: Pre-training text encoders as discriminators rather than generators, 2020.

A. P. Dempster, Laird N. M., and Rubin D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39(1):1–38, 1977.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics (ACL)*, pp. 4171–4186, 2019.

Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv*, 2021.

Zoubin Ghahramani and Michael Jordan. Factorial hidden Markov models. *Machine Learning*, 29: 245–273, 1997.

Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. Hidden topic markov models. In Marina Meila and Xiaotong Shen (eds.), *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pp. 163–170, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR. URL `https://proceedings.mlr.press/v2/gruber07a.html`.

M. Gunst and O. Shcherbakova. Asymptotic behavior of bayes estimators for hidden markov models with application to ion channels. *Mathematical Methods of Statistics*, 17:342–356, 2008.

Keith W. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020.

Ari Holtzman, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. Surface form competition: Why the highest probability answer isn't always right, 2021.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? In *Association for Computational Linguistics (ACL)*, 2020.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Association for Computational Linguistics (ACL)*, 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

B.J.K. Kleijn and A.W. van der Vaart. The Bernstein-von mises theorem under misspecification. *Electronic Journal of Statistics*, 6, 2012.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Association for Computational Linguistics (ACL)*, 2020.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Association for Computational Linguistics (ACL)*, 2021.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. Jurassic-1: Technical details and evaluation. Technical report, AI21 Labs, August 2021.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Denis Paperno, German Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernandez. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Association for Computational Linguistics (ACL)*, 2016.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. Multitask prompted training enables zero-shot task generalization, 2021.

Timo Schick and Hinrich Schütze. Exploiting cloze questions for few shot text classification and natural language inference. In *European Association for Computational Linguistics (EACL)*, 2021.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Eliciting knowledge from language models using automatically generated prompts. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Ingo Steinwart. How to compare different loss functions and their risks. *Constructive Approximation*, 26, 2007.

A. W. van der Vaart. *Asymptotic statistics*. Cambridge University Press, 1998.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

Colin Wei, Sang Michael Xie, and Tengyu Ma. Why do pretrained language models help in downstream tasks? an analysis of head and prompt tuning. *arXiv*, 2021a.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. *arXiv*, 2021b.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. HuggingFace's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning (ICML)*, 2021.

Bernardo Ávila Pires and Csaba Szepesvári. Multiclass classification calibration functions. *arXiv*, 2016.

A Framework details

Prompt distribution details. For in-context learning, we sample a prompt from a new distribution p_{prompt}, which consists of n independent training examples and 1 test example. We first sample n hidden segments H of length k by sampling the first element $h^{\text{start}} = H[1]$ from a prompt start distribution π_{prompt}. Then, we sample the rest of the segment $H^{\text{seg}} = H[2:k]$ from the hidden transition distribution of the pretraining distribution p corresponding to a particular concept θ^*:

$$H_1,\ldots,H_n, \quad H_i = [h_{i,1},\ldots,h_{i,k}] \tag{20}$$

$$h_i^{\text{start}} = H_i[1] \sim \pi_{\text{prompt}}, \quad H_i^{\text{seg}} = H_i[2:k] \sim p(H_i^{\text{seg}} | h^{\text{start}}, \theta^*). \tag{21}$$

To end each example (except the test example), we sample n delimiters $h^{\text{delim}} \in \mathcal{D}$ from $p_{\text{prompt}}^{\text{delim}}$:

$$h_1^{\text{delim}},\ldots,h_n^{\text{delim}}, \quad h_i^{\text{delim}} \sim p_{\text{prompt}}^{\text{delim}}. \tag{22}$$

Conditioned on hidden variables H_i and h_i^{delim}, we sample the observed tokens $O_i = [o_{i,1},\ldots,o_{i,k}]$ and o_i^{delim} respectively from the pre-training distribution:

$$O_1,\ldots,O_n, \quad O_i \sim p(O_i | H_i) \tag{23}$$

$$o_1^{\text{delim}},\ldots,o_n^{\text{delim}}, \quad o_i^{\text{delim}} \sim p(o_i^{\text{delim}} | h_i^{\text{delim}}, \theta^*) \tag{24}$$

The "input" for each example is $x_i = O_i[1:k-1]$ and the "output" is $y_i = O_i[k]$. Taking S to be the sequence of training examples (without the test example), the resulting prompt sequence is

$$[S_n, x_{\text{test}}] = [O_1, o_1^{\text{delim}}, \ldots, O_n, o_n^{\text{delim}}, x_{\text{test}}] = [x_1, y_1, o_1^{\text{delim}}, x_2, y_2, o_2^{\text{delim}}, \ldots, x_n, y_n, o_n^{\text{delim}}, x_{\text{test}}] \sim p_{\text{prompt}} \tag{25}$$

where $x_{\text{test}} = x_{n+1} = O_{n+1}[1:k-1]$ is sampled via the same process but with $k-1$ elements.

B LEMMAS FOR THEOREM 1

The following propositions, which lower bound the probability of a delimiter token and probability of an example under θ^*, are direct corollaries of the assumptions.

Proposition 1. *For all i, we have $p(h_i^{delim}|O_1,o_1^{delim},...,O_i,\theta^*) > c_2$ and $p(h_i^{delim}|O_1,o_1^{delim},...,O_i,\theta) < c_1$.*

Proof. By Assumption 2,

$$p(h_i^{\text{delim}}|O_1,o_1^{\text{delim}},...,O_i,\theta) = \sum_{h_{i,k}} p(h_i^{\text{delim}}|h_{i,k})p(h_{i,k}|O_1,o_1^{\text{delim}},...,O_i,\theta) \tag{26}$$

$$< \sum_{h_{i,k}} c_1 p(h_{i,k}|O_1,o_1^{\text{delim}},...,O_i,\theta) = c_1. \tag{27}$$

Similarly,

$$p(h_i^{\text{delim}}|O_1,o_1^{\text{delim}},...,O_i,\theta^*) = \sum_{h_{i,k}} p(h_i^{\text{delim}}|h_{i,k})p(h_{i,k}|O_1,o_1^{\text{delim}},...,O_i,\theta^*) \tag{28}$$

$$> \sum_{h_{i,k}} c_2 p(h_{i,k}|O_1,o_1^{\text{delim}},...,O_i,\theta^*) = c_2. \tag{29}$$

□

Proposition 2. *The density of an example is lower bounded for θ^*: there is some $c_5 > 0$ such that $p(O_i|h_i^{start},h_{j,l},\theta^*) > c_5$ for all i and future hidden states $h_{j,l}$, for any l and $j > i$.*

Proof. By Assumption 4, we have

$$p(O_i|h_i^{\text{start}},h_{j,l},\theta^*) = \sum_{H_i} p(O_i|H_i)p(H_i|h_i^{\text{start}},h_{j,l},\theta^*) > (c_4)^k \tag{30}$$

for some H_i. We have

$$p(H_i|h_i^{\text{start}},h_{j,l},\theta^*) = \frac{p(h_{j,l}|H,h_i^{\text{start}},\theta^*)p(H|h_i^{\text{start}},\theta^*)}{p(h_{j,l}|h_i^{\text{start}},\theta^*)} > c_3^2 \tag{31}$$

which lower bounds the terms in the numerator by c_3 (marginalizing over previous hidden states), and upper bounding the denominator by 1. Setting $c_5 = (c_4)^k c_3^2$ finishes the proof. □

B.1 MIXING TIME LEMMA

The following lemma is used to show that conditioning on a variable that is at the end of the Markov chain has small effect on variables that are far away from the end of the chain (further than the mixing time of the chain).

Lemma 1. *Suppose $Z_1,...,Z_t$ are sampled from a Markov chain with stationary distribution π and ϵ-mixing time t_{mix}. Suppose A is any random variable where $p(Z_t|Z_i,A) = p(Z_t|Z_i)$. Then for $i < t - t_{mix}$,*

$$p(Z_i|A,Z_t) \in [C_1(\epsilon)p(Z_i|A), C_2(\epsilon)p(Z_i|A)] \tag{32}$$

where

$$C_1(\epsilon) = \frac{\pi(Z_t) - \epsilon}{\pi(Z_t) + \epsilon} \tag{33}$$

$$C_2(\epsilon) = \frac{\pi(Z_t) + \epsilon}{\pi(Z_t) - \epsilon} \tag{34}$$

Proof. Using the definition of mixing time we have

$$|p(Z_t|Z_i) - \pi(Z_t)| < \epsilon, \tag{35}$$

so that

$$p(Z_i,Z_t|A)=p(Z_i|A)p(Z_t|Z_i)\in[p(Z_i|A)(\pi(Z_t)-\epsilon),p(Z_i|A)(\pi(Z_t)+\epsilon)]. \tag{36}$$

Using Bayes' rule,

$$p(Z_i|A,Z_t)=\frac{p(Z_t,Z_i|A)}{p(Z_t)} \tag{37}$$

$$<\frac{p(Z_i|A)(\pi(Z_t)+\epsilon)}{\pi(Z_t)-\epsilon}. \tag{38}$$

using Equation 36 and $p(Z_t)=\mathbb{E}_{Z_1}[p(Z_t|Z_1)]>\pi(Z_t)-\epsilon$ by the definition of mixing time. We also have

$$p(Z_i|A,Z_t)=\frac{p(Z_t,Z_i|A)}{p(Z_t)} \tag{39}$$

$$>\frac{p(Z_i|A)(\pi(Z_t)-\epsilon)}{\pi(Z_t)+\epsilon}. \tag{40}$$

□

Therefore, conditioning on Z_t does not affect the distribution of Z_i much, and as the error ϵ decreases, the effect becomes smaller.

B.2 Lemma for limiting distribution of hidden start state

We aim to show a limiting distribution for the hidden start state. As an intermediate step, we first show lower and upper bounds for the likelihood of the prompt examples given a hidden start state for the test example.

Lemma 2. *Define $\epsilon^\theta(t)$ to be the bound on the TV distance between the distribution on h_{test}^{start} and π^θ for $\epsilon^\theta(t)$-mixing time t. This bound $\epsilon^\theta(t)$ monotonically decreases to 0 as $t\to\infty$. Let $O_i^{ex}=[o_{i-1}^{delim},O_i]$ be the i-th observation segment and the previous delimiter together for $i>1$ and define $O_1^{ex}=O_1$. In the setting of Section 2.1, we have the following upper and lower bounds for any $t\leq n$:*

$$p(S_n,|h_{test}^{start},\theta)\leq p(h_n^{delim}|O_{1:n}^{ex},h_{test}^{start},\theta)$$

$$\prod_{i=1}^{n-t}\left[C_2^\theta(\epsilon^\theta(t))^2\sum_{h_{i-1}^{delim}\in\mathcal{D}}p(O_i|h_{i-1}^{delim},\theta)p(h_{i-1}^{delim}|O_{1:i-1}^{ex},\theta)\right] \tag{41}$$

$$\prod_{i=n-t}^{n}\left[\sum_{h_{i-1}^{delim}\in\mathcal{D}}p(O_i|h_{i-1}^{delim},h_{test}^{start},\theta)p(h_{i-1}^{delim}|O_{1:i-1}^{ex},h_{test}^{start},\theta)\right] \tag{42}$$

$$p(S_n,x_{test}|h_{test}^{start},\theta)\geq p(x_{test}|h_{test}^{start},\theta)p(h_n^{delim}|O_{1:n}^{ex},h_{test}^{start},\theta)$$

$$\prod_{i=1}^{n-t}\left[C_1^\theta(\epsilon^\theta(t))^2\sum_{h_{i-1}^{delim}\in\mathcal{D}}p(O_i|h_{i-1}^{delim},\theta)p(h_{i-1}^{delim}|O_{1:i-1}^{ex},\theta)\right] \tag{43}$$

$$\prod_{i=n-t}^{n}\left[\sum_{h_{i-1}^{delim}\in\mathcal{D}}p(O_i|h_{i-1}^{delim},h_{test}^{start},\theta)p(h_{i-1}^{delim}|O_{1:i-1}^{ex},h_{test}^{start},\theta)\right] \tag{44}$$

$$\tag{45}$$

Proof. We analyze the likelihood term:

$$p(S_n|h_{\text{test}}^{\text{start}},\theta) = p(S_n|h_{\text{test}}^{\text{start}},\theta) \tag{46}$$

$$= p(o_n^{\text{delim}}|O_{1:n}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)\prod_{i=1}^{n} p(O_i^{\text{ex}}|O_{1:i-1}^{\text{ex}}h_{\text{test}}^{\text{start}},\theta) \tag{47}$$

$$= p(o_n^{\text{delim}}|O_{1:n}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta) \prod_{i=1}^{n}\Bigg[\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i^{\text{ex}}|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)\Bigg] \tag{48}$$

$$= p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta) \prod_{i=1}^{n}\Bigg[\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)\Bigg] \tag{49}$$

where in the last step we use our assumption $p(o^{\text{delim}}|h^{\text{delim}}) = 1$ (Assumption 1) for any delimiter token to change O_i^{ex} to O_i and o_n^{delim} to h_n^{delim}. Note that we can take the sum over hidden states to be over $\mathcal{D}$ because of our assumptions on the delimiter hidden states.

First work with

$$p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta) = \sum_{H_i} p(O_i|H_i,h_{i-1}^{\text{delim}},\theta)p(H_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta). \tag{50}$$

Focusing on the last term, we use Lemma 1 to show that for $i < n-t$,

$$p(H_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta) \in [C_1^\theta(\epsilon^\theta(t_n))\cdot p(H_i|h_{i-1}^{\text{delim}},\theta), C_2^\theta(\epsilon^\theta(t))\cdot p(H_i|h_{i-1}^{\text{delim}},\theta)] \tag{51}$$

where $C_1^\theta(\epsilon^\theta(t)), C_2^\theta(\epsilon^\theta(t))$ are as defined in Lemma 1, and both quantities converge to 1 as $\epsilon(t) \to 0$. Thus, for $i < n-t$,

$$p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta) \le C_2^\theta(\epsilon^\theta(t))\cdot p(O_i|h_{i-1}^{\text{delim}},\theta). \tag{52}$$

For the other term, we also invoke Lemma 1 to show that for $i < n-t$

$$p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta) \le C_2^\theta(\epsilon^\theta(t))\cdot p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta). \tag{53}$$

We can put these together to bound the terms in the original sum (Equation 49) for $i < n-t$:

$$\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta) \tag{54}$$

$$\le C_2^\theta(\epsilon^\theta(t))^2 \sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta). \tag{55}$$

While these steps result in an upper bound, using $C_1^\theta(\epsilon^\theta(t))$ yields the lower bound. □

The following lemma is used to show that the hidden start state distribution converges to the stationary distribution with infinite examples in the prompt.

Lemma 3. *In the setting of Section 2.1, we have as $n \to \infty$ that*

$$p(h_{test}^{start}|S_n,x_{test},\theta) \to \pi^\theta(h_{test}^{start}|x_{test}). \tag{56}$$

Proof. First, by Bayes' rule, we have

$$p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta) = \frac{p(S_n,x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}}|\theta)}{p(S_n,x_{\text{test}}|\theta)} \tag{57}$$

$$= \frac{p(S_n|h_{\text{test}}^{\text{start}},\theta)p(x_{\text{test}}|h_{\text{test}}^{\text{start}})p(h_{\text{test}}^{\text{start}}|\theta)}{p(S_n,x_{\text{test}}|\theta)}. \tag{58}$$

We will analyze the following ratio for $h_{\text{test}}^{\text{start}}$ where $p(S_n|h_{\text{test}}^{\text{start}},\theta) > 0$ and show it converges to a constant wrt $h_{\text{test}}^{\text{start}}$:

$$\frac{p(S_n,|\, h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n,|\, h_{\text{test}}^{\text{start}},\theta)} \to \Theta(1) \tag{59}$$

which implies that

$$\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n|h_{\text{test}}^{\text{start}},\theta)p(x_{\text{test}}|h_{\text{test}}^{\text{start}})} = \frac{\sum_{h_{\text{test}}^{\text{start}\prime}} p(S_n|h_{\text{test}}^{\text{start}\prime},\theta)p(x_{\text{test}}|h_{\text{test}}^{\text{start}\prime})p(h_{\text{test}}^{\text{start}\prime}|\theta)}{p(S_n|h_{\text{test}}^{\text{start}},\theta)p(x_{\text{test}}|h_{\text{test}}^{\text{start}})} \tag{60}$$

$$\to \Theta(1)\frac{p(x_{\text{test}}|\theta)}{p(x_{\text{test}}|h_{\text{test}}^{\text{start}})}. \tag{61}$$

Plugging this into Equation 58 (after taking the reciprocal) and noting that $p(h_{\text{test}}^{\text{start}}|\theta) \to \pi^\theta(h_{\text{test}}^{\text{start}})$ show that

$$p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta) \to \Theta(1)\frac{p(x_{\text{test}}|h_{\text{test}}^{\text{start}})\pi^\theta(h_{\text{test}}^{\text{start}})}{p(x_{\text{test}}|\theta)} \tag{62}$$

$$= \Theta(1)\pi^\theta(h_{\text{test}}^{\text{start}}|x_{\text{test}}). \tag{63}$$

Note that we do not take the limit of the x_{test} terms since x_{test} is observed. Since the limiting distribution is a valid distribution (the simplex is closed) and should sum to 1, we can apply this constraint to show that the constant factor is 1, which shows the result.

Now it remains to show that

$$\frac{p(S_n,|\, h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n,|\, h_{\text{test}}^{\text{start}},\theta)} \to \Theta(1). \tag{64}$$

Redefining the ratio, we have

$$\frac{p(S_n,|\, h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n,|\, h_{\text{test}}^{\text{start}},\theta)} = \exp\left(n \cdot \frac{1}{n}\log\frac{p(S_n\,|\, h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n\,|\, h_{\text{test}}^{\text{start}},\theta)}\right). \tag{65}$$

As in Lemma 2, let $O_i^{\text{ex}} = [o_{i-1}^{\text{delim}}, O_i]$ be the i-th observation segment and the previous delimiter together for $i > 1$ and define $O_1^{\text{ex}} = O_1$. Define $\epsilon^\theta(t)$ to be the bound on the TV distance between the distribution on $h_{\text{test}}^{\text{start}}$ vs. π^θ, for $\epsilon^\theta(t)$-mixing time t. Using Lemma 2, we have the following upper and lower bounds:

$$\frac{1}{n}\log\frac{p(S_n\,|\, h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n\,|\, h_{\text{test}}^{\text{start}},\theta)} \tag{66}$$

$$\leq \frac{1}{n}\left(\sum_{i=1}^{n-t}\left[\log\frac{C_2^\theta(\epsilon^\theta(t))^2}{C_1^\theta(\epsilon^\theta(t))^2} + \log\frac{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta)}{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta)}\right]\right.$$

$$\left. + \log\frac{p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},h_{\text{test}}^{\text{start}\prime},\theta)}{p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)} + \sum_{i=n-t}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}\prime},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}\prime},\theta)}{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},h_{\text{test}}^{\text{start}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)}\right)$$

$$\leq \frac{1}{n}\left(\sum_{i=1}^{n-t}\log\frac{C_2^\theta(\epsilon^\theta(t))^2}{C_1^\theta(\epsilon^\theta(t))^2} + (\log(c_1)-\log(c_2)) + \sum_{i=n-t}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} 1\cdot p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}\prime},\theta)}{\sum_{h_{i-1}^{\text{delim}}\in\mathcal{D}} c_5 p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},h_{\text{test}}^{\text{start}},\theta)}\right)$$

$$\leq \frac{1}{n}\left(\sum_{i=1}^{n-t}\log\frac{C_2^\theta(\epsilon^\theta(t))^2}{C_1^\theta(\epsilon^\theta(t))^2} + (\log(c_1)-\log(c_2)) - \sum_{i=n-t}^{n}\log(c_5)\right)$$

$$\leq \frac{1}{n}\left((\log(c_1)-\log(c_2)) + O(t)\right) \tag{67}$$

$$= O(1/n) \tag{68}$$

We used both Propositions 1 and 2 in the terms involving c_1, c_2 and c_5. Note that in the first line, the sum can must be over the set of delimiter states $\mathcal{D}$ by using the assumption that observing a

delimiter o^{delim} implies that the corresponding hidden state h^{delim} must be in $\mathcal{D}$. where we note that $t, C_2^\theta(\epsilon^\theta(t)), C_1^\theta(\epsilon^\theta(t))$ are constants. The first of the two log ratios involving a marginalization over h_{i-1}^{delim} is 0 since the numerator and denominator are the same. Flipping the order of $C_2^\theta(\epsilon^\theta(t))$,$C_1^\theta(\epsilon^\theta(t))$, c_1,c_2 and 1 vs. c_5 gives a lower bound:

$$\frac{1}{n}\log\frac{p(S_n,|\,h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n,|\,h_{\text{test}}^{\text{start}},\theta)} \geq \frac{1}{n}\left(\sum_{i=1}^{n-t}\log\frac{C_1^\theta(\epsilon^\theta(t))^2}{C_2^\theta(\epsilon^\theta(t))^2} + (\log(c_2)-\log(c_1)) + \sum_{i=n-t}^{n}\log(c_5)\right)$$

$$\geq \frac{1}{n}\left((\log(c_2)-\log(c_1))+\Omega(t)\right) \tag{69}$$

$$= \Omega(-1/n) \tag{70}$$

Together, this shows that

$$\frac{p(S_n\,|\,h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n\,|\,h_{\text{test}}^{\text{start}},\theta)} = \exp\left(n\cdot\frac{1}{n}\log\frac{p(S_n\,|\,h_{\text{test}}^{\text{start}\prime},\theta)}{p(S_n\,|\,h_{\text{test}}^{\text{start}},\theta)}\right) = \Theta(1) \tag{71}$$

as desired. □

C Proof of Theorem 1

Proof. We analyze the most likely prediction over the pretraining distribution conditioned on the prompt $\text{argmax}_y p(y|S_n,x_{\text{test}})$.

$$p(y|S_n,x_{\text{test}}) = \int_\theta p(y,|S_n,x_{\text{test}},\theta)p(\theta|S_n,x_{\text{test}})d\theta \tag{72}$$

$$\propto \int_\theta p(y,|S_n,x_{\text{test}},\theta)p(S_n,x_{\text{test}}|\theta)p(\theta)d\theta \tag{73}$$

$$\propto \int_\theta p(y,|S_n,x_{\text{test}},\theta)\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}p(\theta)d\theta \tag{74}$$

$$= \int_\theta \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta)\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}p(\theta)d\theta \tag{75}$$

Defining the following quantity,

$$r_n(\theta) = \frac{1}{n}\log\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}. \tag{76}$$

we will show that under distinguishability for all $\theta\neq\theta^*$, $r_n(\theta)$ converges to a negative constant such that

$$\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)} = \exp(n\cdot r_n(\theta)) = 0 \tag{77}$$

for $\theta\neq\theta^*$, whereas this ratio is always 1 for $\theta=\theta^*$. This will then "select" the desired prompt concept through marginalization.

Taking the limit of Equation 75, we have

$$\lim_{n\to\infty}\int_\theta \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta)\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}p(\theta)d\theta \tag{78}$$

$$= \lim_{n\to\infty}\int_\theta \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta)\exp(n\cdot r_n(\theta))p(\theta)d\theta \tag{79}$$

$$= \int_\theta \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} \lim_{n\to\infty} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}}|S_n,x_{\text{test}},\theta)\exp(n\cdot r_n(\theta))p(\theta)d\theta \tag{80}$$

$$= \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta^*)\pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}})p(\theta^*) \tag{81}$$

$$\propto \sum_{h_{\text{test}}^{\text{start}}\in\mathcal{H}} p(y,|S_n,x_{\text{test}},h_{\text{test}}^{\text{start}},\theta^*)\pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}}) \tag{82}$$

$$= \mathbb{E}_{h_{\text{test}}^{\text{start}}\sim\pi^{\theta^*}(h_{\text{test}}^{\text{start}}|x_{\text{test}})}\left[p(y|x_{\text{test}},h_{\text{test}}^{\text{start}},\theta^*)\right] \tag{83}$$

as desired, where we can swap the limit and integral by the dominated convergence theorem (probabilities are bounded above by 1, and $p(\theta)$ are bounded above by Assumption 4), and we used Lemma 3 to show the convergence to the stationary distribution.

Now it remains to show that $r_n(\theta)$ converges to a negative constant for $\theta \neq \theta^*$. Let $O_i^{\text{ex}} = [o_{i-1}^{\text{delim}}, O_i]$ be the i-th observation segment and the previous delimiter together for $i > 1$ and define $O_1^{\text{ex}} = O_1$. Expanding the numerator of the ratio in $r_n(\theta)$, we have

$$p(S_n, x_{\text{test}}|\theta) = p(x_{\text{test}}|S_n,\theta)p(S_n|\theta) \tag{84}$$

$$= \sum_{h_{\text{test}}^{\text{start}}} p(x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}} \mid S_n,\theta)p(o_n^{\text{delim}}|O_{1:n}^{\text{ex}},\theta)\prod_{i=1}^{n} p(O_i^{\text{ex}}|O_{1:i-1}^{\text{ex}},\theta) \tag{85}$$

$$= \sum_{h_{\text{test}}^{\text{start}}} p(x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}} \mid S_n,\theta) \tag{86}$$

$$\sum_{h_n^{\text{delim}} \in \mathcal{D}} p(o_n^{\text{delim}}|h_n^{\text{delim}})p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},\theta)\prod_{i=1}^{n}\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta) \tag{87}$$

$$= \sum_{h_{\text{test}}^{\text{start}}} p(x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}} \mid S_n,\theta) \tag{88}$$

$$\sum_{h_n^{\text{delim}} \in \mathcal{D}} p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},\theta)\prod_{i=1}^{n}\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta) \tag{89}$$

Note that in the last line, the sum must be over the set of delimiter states $\mathcal{D}$ by using the assumption that observing a delimiter o^{delim} implies that the corresponding hidden state h^{delim} must be in $\mathcal{D}$.

We restrict our attention to θ where $p(S_n, x_{\text{test}}|\theta) > 0$, since otherwise θ does not affect the prediction. Expanding $r_n(\theta)$, we have the following upper bound:

$$r_n(\theta) = \frac{1}{n}\left(\log\frac{p(S_n, x_{\text{test}}|\theta)}{p(S_n, x_{\text{test}}|\theta^*)}\right) \tag{90}$$

$$= \frac{1}{n}\left(\log\frac{\sum_{h_{\text{test}}^{\text{start}}} p(x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta)p(h_{\text{test}}^{\text{start}} \mid S_n,\theta)}{\sum_{h_{\text{test}}^{\text{start}}} p(x_{\text{test}}|h_{\text{test}}^{\text{start}},\theta^*)p(h_{\text{test}}^{\text{start}} \mid S_n,\theta^*)} + \log\frac{\sum_{h_n^{\text{delim}} \in \mathcal{D}} p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},\theta)}{\sum_{h_n^{\text{delim}} \in \mathcal{D}} p(h_n^{\text{delim}}|O_{1:n}^{\text{ex}},\theta^*)}\right. \tag{91}$$

$$\left. + \sum_{i=1}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta)}{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta^*)p(h_{i-1}^{\text{delim}}|O_{1:i-1}^{\text{ex}},\theta^*)}\right) \tag{92}$$

$$\leq \frac{1}{n}\left(\log\frac{\sum_{h_{\text{test}}^{\text{start}}} 1 \cdot p(h_{\text{test}}^{\text{start}} \mid S_n,\theta)}{\sum_{h_{\text{test}}^{\text{start}}} c_5 \cdot p(h_{\text{test}}^{\text{start}} \mid S_n,\theta^*)} + n(\log(c_1) - \log(c_2)) + \sum_{i=1}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)}{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta^*)}\right) \tag{93}$$

$$= \frac{1}{n}\left(-\log(c_5) + n(\log(c_1) - \log(c_2)) + \sum_{i=1}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta)}{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(O_i|h_{i-1}^{\text{delim}},\theta^*)}\right) \tag{94}$$

$$= \frac{1}{n}\left(-\log(c_5) + n(\log(c_1) - \log(c_2)) + \sum_{i=1}^{n}\log\frac{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(h_{i-1}^{\text{delim}}|\theta)p(O_i|h_{i-1}^{\text{delim}},\theta)}{\sum_{h_{i-1}^{\text{delim}} \in \mathcal{D}} p(h_{i-1}^{\text{delim}}|\theta^*)p(O_i|h_{i-1}^{\text{delim}},\theta^*)}\right) \tag{95}$$

$$= \frac{1}{n}\left(-\log(c_5) + n(\log(c_1) - \log(c_2)) + \sum_{i=1}^{n}\log\frac{p(O_i|\theta)}{p(O_i|\theta^*)}\right) \tag{96}$$

$$\to_{n\to\infty} \mathbb{E}_{O\sim p_{\text{prompt}}}\left[\log\frac{p(O|\theta)}{p(O|\theta^*)}\right] + \log(c_1) - \log(c_2) \tag{97}$$

We used both Propositions 1 and 2 in the terms involving c_1, c_2 (bounding the probability of h^{delim} hidden states) and c_5 (bounding the probability of x_{test}). Note that in the second line, the sum can

must be over the set of delimiter states $\mathcal{D}$ by using the assumption that observing a delimiter o^{delim} implies that the corresponding hidden state h^{delim} must be in $\mathcal{D}$. In the second to last line, we used the assumption that $p(h^{\text{delim}}|\theta)=\frac{1}{|\mathcal{D}|}$ for all θ,h^{delim}, multiplying both top and bottom of the ratio by this constant. The expectation becomes a KL divergence for token 2 through k of each segment (the first token is different due to the starting distribution π_{prompt}).

Aiming to decompose the expectation into a sum over the k tokens, we write $p^j_\theta(o)=p(O[j]=o|O[1:j-1],\theta)$. Then we have that

$$\lim_{n\to\infty} r_n(\theta) < -\sum_{j=2}^{k}\mathbb{E}_{O[1:j-1]\sim p_{\text{prompt}}}[KL(p^j_{\theta^*}\|p^j_\theta)]+\mathbb{E}_{O[1]\sim p_{\text{prompt}}}\left[\log\frac{p(O[1]|\theta)}{p(O[1]|\theta^*)}\right]+(\log(c_1)-\log(c_2)) \tag{98}$$

The second term is an error term that depends on how different the starting prompt distribution π_{prompt} (which is part of p_{prompt}) is to the pre-training distribution. The third term is an error term that comes from the delimiter transitions. The bound is negative when the sum of KL terms is larger in magnitude than the error terms. Note that as k becomes larger, the number of observations of θ^* "overpowers" the OOD transitions in the prompt distribution. This condition is equivalent to the disinguishability condition (Condition 1).

By assumption, for $\theta\neq\theta^*$ the Condition 1 holds, and thus

$$\lim_{n\to\infty}\frac{p(S_n,x_{\text{test}}|\theta)}{p(S_n,x_{\text{test}}|\theta^*)}=\lim_{n\to\infty}\exp(n\cdot r_n(\theta))=0 \tag{99}$$

since $r_n(\theta)$ has a negative, constant limit. Note that $\exp(n\cdot r_n(\theta^*))=1$ for θ^*.

□

D NON-DISTINGUISHABLE CASE

When Condition 1 is unsatisfied, Equation 14), gives an upper bound on the sum of KL divergences for the next token distributions given different-length histories. In contrast, the in-context task only measures the accuracy of the last (k-th) token. The main challenge is to relate the different-length histories to each other to give a more precise bound for the error on the in-context task (last token).

Before addressing this challenge, we give the following lemma, which leverages the result of Ávila Pires & Szepesvári (2016); Steinwart (2007) to relate a bound on the KL divergence to 0-1 loss.

Lemma 4. *Let the set of θ which does not satisfy Condition 1 to be $\mathcal{B}$. Assume that $KL(p(y_{test}|x_{test},\theta^*)\|p(y_{test}|x_{test},\theta)$ is bounded above for all θ. If*

$$\mathbb{E}_{x_{test}\sim p_{prompt}}[KL(p(y_{test}|x_{test},\theta^*)\|p(y_{test}|x_{test},\theta))]\leq\epsilon_\theta \quad \textit{for all} \quad \theta\in\mathcal{B}, \tag{100}$$

then

$$\lim_{n\to\infty}L_{0\text{-}1}(f_n)\leq\inf_f L_{0\text{-}1}(f)+g^{-1}\left(\sup_{\theta\in\mathcal{B}}\epsilon_\theta\right) \tag{101}$$

where

$$g(\delta)=\frac{1}{2}((1-\delta)\log(1-\delta)+(1+\delta)\log(1+\delta)) \tag{102}$$

is a calibration function for the multiclass logistic loss for $\delta\in[0,1]$.

Proof. First, we note that we can study the 0-1 risk of the limiting predictor:

$$\lim_{n\to\infty}L_{0\text{-}1}(f_n)=\lim_{n\to\infty}\mathbb{E}_{x_{\text{test}},y_{\text{test}}\sim p_{\text{prompt}}}[\mathbf{1}[f_n(x_{\text{test}})\neq y_{\text{test}}]] \tag{103}$$

$$=\mathbb{E}_{x_{\text{test}},y_{\text{test}}\sim p_{\text{prompt}}}[\lim_{n\to\infty}\mathbf{1}[f_n(x_{\text{test}})\neq y_{\text{test}}]] \text{ (dominated convergence, boundedness of indicator)} \tag{104}$$

$$=\mathbb{E}_{x_{\text{test}},y_{\text{test}}\sim p_{\text{prompt}}}[\mathbf{1}[\lim_{n\to\infty}f_n(x_{\text{test}})\neq y_{\text{test}}]] \tag{105}$$

where in the last step we use that since the output space of f_n is discrete and the probabilities that the in-context predictor takes an argmax over converges, then for N large enough, $f_N(x_{\text{test}}) = \lim_{n\to\infty} f_n(x_{\text{test}})$.

Note that for every input x_{test}, the limiting in-context learning predictor outputs the argmax of a predictive distribution which can be a mixture of predictive distributions over $\mathcal{B}$:

$$\lim_{n\to\infty} f_n(x_{\text{test}}) = \operatorname*{argmax}_y \mathbb{E}_{\theta\sim q}[p(y|x_{\text{test}},\theta)] \tag{106}$$

for some distribution q over $\mathcal{B}$. The KL divergence between this mixture and the prompt concept is bounded by the KL divergence of any one $\theta \in \mathcal{B}$, due to the convexity of KL:

$$\begin{aligned}
&\mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y|x_{\text{test}},\theta^*)\|\mathbb{E}_{\theta\sim q}[p(y|x_{\text{test}},\theta)]] && (107)\\
&\leq \mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[\mathbb{E}_{\theta\sim q}[KL(p(y|x_{\text{test}},\theta^*)\|p(y|x_{\text{test}},\theta))]] && (108)\\
&= \mathbb{E}_{\theta\sim q}[\mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y|x_{\text{test}},\theta^*)\|p(y|x_{\text{test}},\theta))]] && (109)\\
&\leq \sup_{\theta\in\mathcal{B}} \mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y|x_{\text{test}},\theta^*)\|p(y|x_{\text{test}},\theta))] && (110)
\end{aligned}$$

where we can exchange the order of expectations since the KL is bounded (dominated convergence).

From the KL bound $KL(p(y_{\text{test}}|x_{\text{test}},\theta^*)\|p(y_{\text{test}}|x_{\text{test}},\theta)$, we thus have

$$\mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y_{\text{test}}|x_{\text{test}},\theta^*)\|p(y_{\text{test}}|x_{\text{test}},\theta))] = L_{\text{CE}}(\theta) - L_{\text{CE}}(\theta^*) \leq \sup_{\theta\in\mathcal{B}} \epsilon_\theta \tag{111}$$

where $L_{\text{CE}}(\theta) = -\mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[p(y_{\text{test}}|x_{\text{test}},\theta^*)\log p(y_{\text{test}}|x_{\text{test}},\theta)]$ is the multiclass logistic risk, and $L_{\text{CE}}(\theta^*)$ is the entropy of the predictive distribution. Applying Theorem 2.2 and 5.11 of Ávila Pires & Szepesvári (2016), g is a calibration function for the multiclass logistic loss, and allows us to convert the surrogate risk bound to a bound on the 0-1 loss, giving the result. Note that we have zero approximation error here, since $\theta^* \in \Theta$. □

Note that g^{-1} is roughly linear in ϵ for ϵ smaller than 0.7, where the bound is non-vacuous.

D.1 Proof of Theorem 2

Proof. By the continuity assumption, we have for any θ in $\mathcal{B}$ that

$$\begin{aligned}
\sum_{j=2}^{k} KL_j(\theta^*\|\theta) &\geq \frac{1}{2}\sum_{j=2}^{k}(\theta-\theta^*)^\top I_{j,\theta^*}(\theta-\theta^*) + (k-1)O(\|\theta-\theta^*\|^3) && (112)\\
&\geq \frac{1}{2}(k-1)\lambda_{\min}(I_{j,\theta^*})\|\theta-\theta^*\|^2 && (113)\\
\implies \|\theta-\theta^*\|^2 &\leq \frac{\epsilon^\theta_{\text{start}} + \epsilon^\theta_{\text{delim}}}{\frac{1}{2}(k-1)(\min_j \lambda_{\min}(I_{j,\theta^*}))}. && (114)
\end{aligned}$$

To bound the last KL term,

$$\begin{aligned}
KL_k(\theta^*\|\theta) &= \frac{1}{2}\sum_{j=2}^{k}(\theta-\theta^*)^\top I_{j,\theta^*}(\theta-\theta^*) + O(\|\theta-\theta^*\|^3) && (115)\\
&\leq \frac{1}{2}(\max_j \lambda_{\max}(I_{j,\theta^*}))\|\theta-\theta^*\|^2 + O(\|\theta-\theta^*\|^2) && (116)\\
&\leq \frac{(\epsilon^\theta_{\text{start}} + \epsilon^\theta_{\text{delim}})(\max_j \lambda_{\max}(I_{j,\theta^*}) + O(1))}{(k-1)\min_j \lambda_{\min}(I_{j,\theta^*})}. && (117)
\end{aligned}$$

Rearranging and noting that $KL_k(\theta^*\|\theta) = \mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y_{\text{test}}|x_{\text{test}},\theta^*)\|p(y_{\text{test}}|x_{\text{test}},\theta))]$, we have

$$\mathbb{E}_{x_{\text{test}}\sim p_{\text{prompt}}}[KL(p(y_{\text{test}}|x_{\text{test}},\theta^*)\|p(y_{\text{test}}|x_{\text{test}},\theta))] \leq \frac{(\epsilon^\theta_{\text{start}} + \epsilon^\theta_{\text{delim}})(\max_j \lambda_{\max}(I_{j,\theta^*}) + O(1))}{(k-1)\min_j \lambda_{\min}(I_{j,\theta^*})} \tag{118}$$

Plugging into Lemma 4 gives the result. □

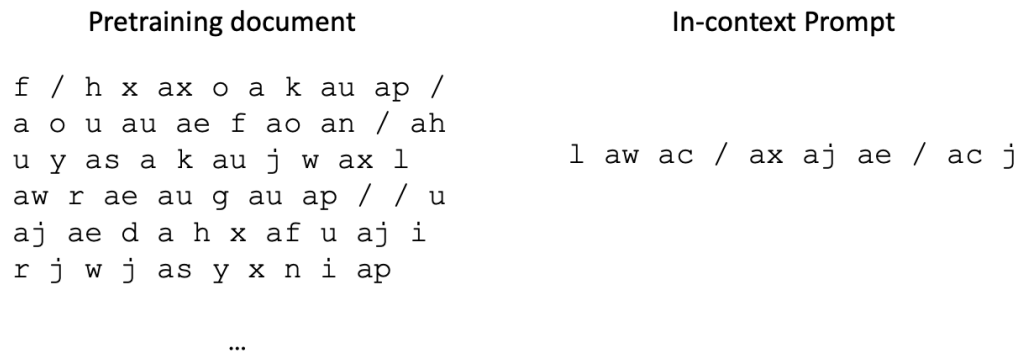

Figure 7: Example pretraining document snippet (**Left**) and example prompt with 3 training examples, 1 test example, and example length 3 (**Right**). The delimiter token is the backslash.

D.2 Proof of Theorem 3

Note that Condition 1 ensures that the sum of KL divergences between positions within a k-length input is bounded. This means that we have a bound over not only the last-position KL divergence, but also for all the intermediate tokens. Intuitively, the random length test example allows the in-context predictor to "take credit" for fitting the intermediate tokens. The proof is immediate given the KL bound and Lemma 4, given that the length of x_{test} is uniformly random between 2 to k.

Proof. Let the set of θ that does not satisfy Condition 1 to be $\mathcal{B}$. We have for any θ in $\mathcal{B}$ that

$$\mathbb{E}_{x_{\text{test}} \sim p_{\text{prompt}}} KL(p(y_{\text{test}}|x_{\text{test}},\theta^* \| p(y_{\text{test}}|x_{\text{test}},\theta)) \tag{119}$$

$$\leq \frac{1}{k-1} \sum_{j=2}^{k} \mathbb{E}_{O[1:j-1] \sim p_{\text{prompt}}} KL(p(O[j]|O[1:j-1],\theta^* \| p(O[j]|O[1:j-1],\theta) \tag{120}$$

$$\leq \frac{\sup_\theta(\epsilon^\theta_{\text{start}} + \epsilon^\theta_{\text{delim}})}{k-1} \tag{121}$$

by Theorem 1 and Condition 1. Plugging this into Lemma 4 gives the result. □

E Experimental details

E.1 GINC dataset

The GINC dataset is generated from a mixture of HMMs. These HMMs output tokens from a vocabulary of size in $\{50,100,150\}$. The vocabulary contains a special delimiter token (backslash – see Figure **??**, designated to be index 1. The vocabulary is generated as combinations of letters starting from a to z, then aa to az, and so on. All sequences are tokenized by splitting on whitespaces.

Memory matrix. The shared memory matrix has 10 entities and 10 properties, totaling 100 entries (corresponding to 100 hidden states). The first column of the memory matrix is fixed to be the delimiter token, while each remaining entry of the shared memory matrix is populated with a token sampled uniformly from the vocabulary.

Transition matrix for properties. We generate 5 property transition matrices, one for each component of the HMM mixture. We generate each transition matrix via a convex combination of 100 random permutation matrices. The weights of the convex combination are randomly generated as

$$\text{softmax}((u-0.5)/t) \tag{122}$$

where $u \in \mathbb{R}^{100}$ has uniform random entries in $[0,1]$ and t is a temperature parameter, set to 0.1.

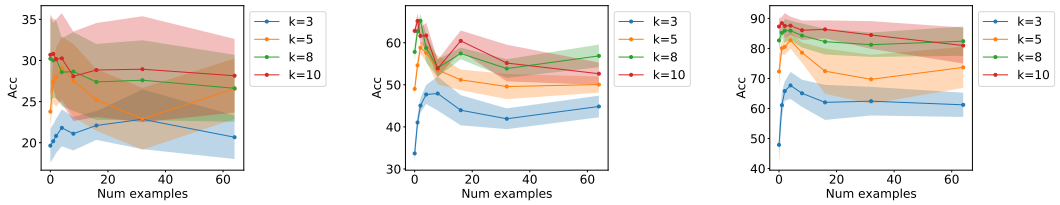

Figure 8: In-context accuracy curve of the 4 layer Transformer on the GINC dataset when the entity transition matrix does not have an additional identity component, for vocabulary sizes 50 (left), 100 (middle), and 150 (right). In-context learning is still generally successful.

Transition matrix for entities. The entity transition matrix is shared between all the HMMs that consistute the mixture. The entity transition matrix is generated in the same way as the property transition matrices, except with one additional step. Letting T be a transition matrix sampled in the same way as a property transition matrix,

In pretraining documents, we expect that the entities (e.g., Einstein) change slowly over time while and the properties of the entity (e.g., their occupation) change quickly with some pattern to generate natural sentences. We implement this by ensuring that the probability of transitioning to the same entity index in the next step is at least 0.9. The final entity transition matrix is then $0.1T+0.9I$ where I is the identity matrix. Although we add the diagonal component for added realism, we also consider not adding this component. Figure 8 shows in-context learning curves for a small (4 layer) Transformer trained on data that does not add the diagonal component (we check this for vocabulary sizes 50, 100, and 150). In-context learning still works in this case, although not as well for the 50 vocab size case.

Start distribution. The starting distribution for the hidden states in all HMMs in the mixture are close to uniform. We generate the start distribution as $\text{softmax}((u-0.5)/t)$ for random vector u with entries uniformly from $[0,1]$ and temperature $t=10$. In the pretraining documents, we only sample from the start distribution in the beginning of the document.

Prompt distribution. The prompts are sampled according to the theoretical setup. The prompt start distribution for each prompt is chosen to deterministically output one property that was randomly chosen to be the starting property for each example in the prompt. Each prompt has an independently sampled starting property. Each prompt also generates examples from one particular HMM in the mixture (which HMM this is is chosen randomly). Given the starting property, we sample k tokens from the HMM. Finally, we append the delimiter token for the example. We repeat this process for each example in the prompt, concatenating all examples. The label is generated as

$$\underset{y}{\text{argmax}}\, p(y|x_{\text{test}},\theta^*) \tag{123}$$

under the prompt concept θ^*. This differs from the theory, which samples y_{test} instead of taking it to be the most likely token. However, there can be a large amount of intrinsic error that sampling introduces. We define the label this way in the simulations to remove the intrinsic error from sampling.

Example of prompt generation. In the example in Figure 7 (right), the starting property is fixed to be 5 (for example). The first token (l) is generated by sampling a random entity index (3), and indexing into the memory matrix returns l. Running the hidden state chain of the HMM forward gives the next pair of property and entity. Since the entity Markov chain changes slowly, the entity is still 3 in the next step – however, the property has changed to 4, and indexing into the memory matrix outputs the next token (aw). Following this same process to generate the third token (the output for the first example), we finish generating one example. To end the example, we append a delimiter (backslash). We repeat this example generation process for all the examples, except for the test example at the end, where we do not generate the last token. We condition the HMM on the generated prompt to compute the posterior distribution over the next token $p(y|x_{\text{test}},\theta^*)$. We take the argmax of this distribution to be the ground truth label.

Dataset details. The dataset contains 1000 training documents and 100 validation documents, where training documents have 10240 tokens and validation documents have 1024 tokens. Each document

is generated by first selecting one of the HMMs from the mixture uniformly at random, then generating 10240 tokens from the HMM.

We also generate 2500 in-context prompts for each (example length,number of examples) pair, for example lengths $k = [3,5,8,10]$ and number of examples $n = [0,1,2,4,8,16,32,64]$. Each prompt is generated using a random HMM in the mixture.

E.2 Transformer details

Our Transformer models are based on the GPT-2 architectures with 4, 12, and 16 layers respectively, with 12 attention heads, 768 dimensional embeddings, residual/embedding/attention dropout set to 0.1, and a context window of 1024. Other than the number of layers, the other parameters are the default settings from the HuggingFace library (Wolf et al., 2019). We train for 5 epochs using the AdamW optimizer (Loshchilov & Hutter, 2019; Kingma & Ba, 2015) with a batch size of 8 and a linear learning rate schedule (with 1000 step warmup) up to a learning rate of 8e-4 for the 4 layer and 12 layer model, while for the 16 layer model we start with a constant learning rate of 8e-4 and reduce by a factor of 0.25 whenever the best validation loss does not improve. We tried both learning rate strategies for all models and take the most consistent. We tuned these models so that the training loss curves between seeds have smaller variability between the runs in terms of the curve shape and when the loss decreases – we found that this is an important indication of stable results. The models took 50 minutes, 2 hours, 3 hours to train respectively. The hardware was mainly Titan Xp GPUs, trained and evaluated using 16-bit precision. All the results are reported with 5 pretraining runs (5 different seeds).

E.3 LSTM details

We train an LSTM language model with embedding size 768, hidden layer size 768, and 6 layers. We use dropout 0.2 and weight decay 1e-5. The optimizer is AdamW starting with a learning rate of 1e-3, then reducing by a factor of 0.25 whenever the best validation loss does not go down. We train for a total of 10 epochs, with gradient clipping at norm 1.0. We use a batch size of 8 and backpropagate through time for 1024 steps (each pretraining data segment is also 1024 tokens). Each model takes roughly 2 hours to train on Titan Xp GPUs.

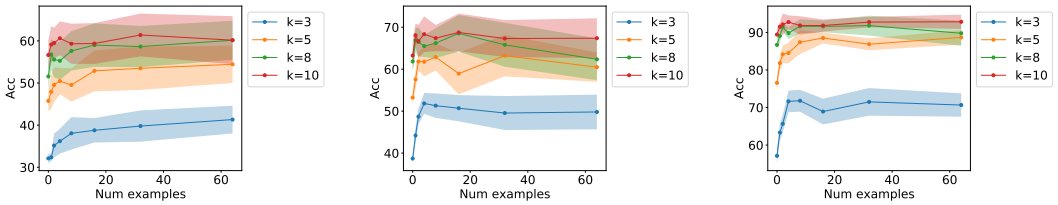

Figure 9: In-context accuracy of the 4 layer Transformer on the GINC dataset for vocabulary sizes 50 (left), 100 (middle) and 150 (right). Accuracies generally improve as the vocabulary size increases.

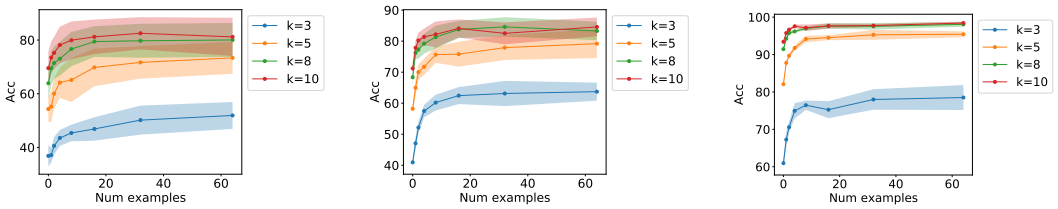

Figure 10: In-context accuracy of the 12 layer Transformer on the GINC dataset for vocabulary sizes 50 (left), 100 (middle) and 150 (right). Accuracies generally improve as the vocabulary size increases.

E.4 Varying the vocabulary size

To do well on the in-context learning task, the model must both infer the prompt concept and the last HMM hidden state. In general, increasing the number of observable symbols makes the in-context task easier by making the inference of the HMM hidden state easier. With more symbols, each hidden state is more likely to output a different symbol, making the inference problem easier. This improvement comes despite the number of output classes in the problem (same as the vocabulary size) increasing.

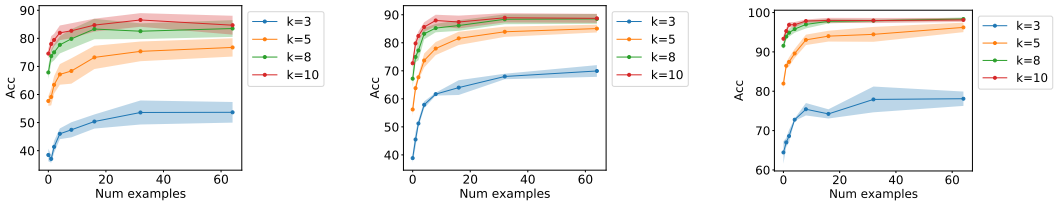

Figure 11: In-context accuracy of the 16 layer Transformer on the GINC dataset for vocabulary sizes 50 (left), 100 (middle) and 150 (right). Accuracies generally improve as the vocabulary size increases.

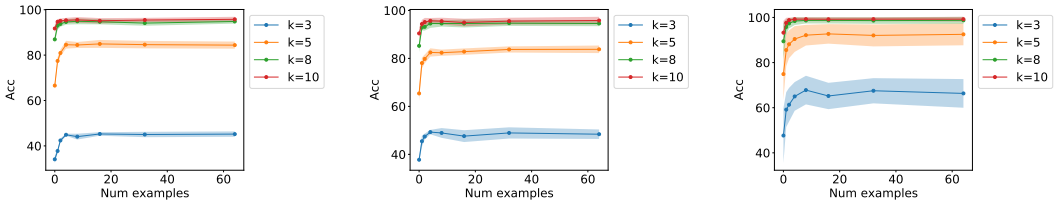

Figure 12: In-context accuracy of the LSTM on the GINC dataset for vocabulary sizes 50 (left), 100 (middle) and 150 (right). Accuracies generally improve as the vocabulary size increases.

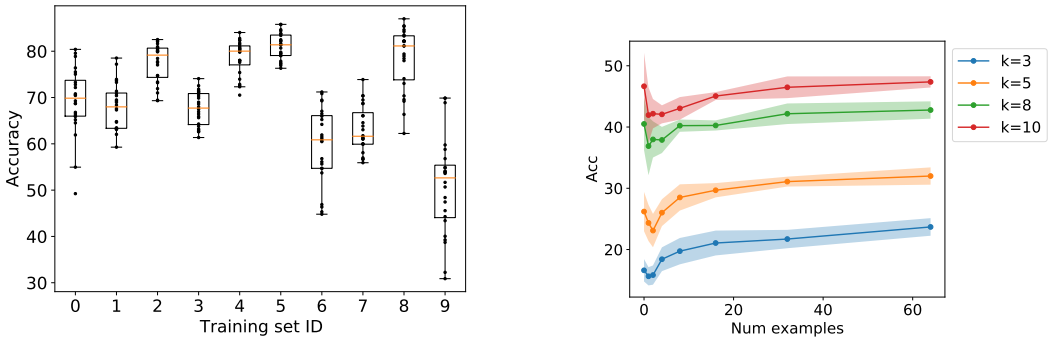

Figure 13: **(Left)** In-context accuracy varies widely with example ordering. Each training ID refers to a set of training examples. Each dot refers to the in-context learning accuracy of one permutation of the training examples for that particular training ID. **(Right)** Zero-shot performance can be higher than one/few-shot performance in some settings in GINC, mirroring the behavior of GPT-3 on some datasets such as LAMBADA (Brown et al., 2020). The few-shot setting introduces the OOD prompt structure, which can initially lower accuracy.

Figures 9, 10, 11, 12 show in-context learning curves for vocabulary sizes 50, 100, and 150, keeping other hyperparmeters of the dataset the same.

E.5 Sensitivity to example ordering

In Figure 13 (left), we test the sensitivity of the in-context accuracy on GINC to the ordering of the prompt examples, following Zhao et al. (2021). For this experiment, we consider prompts generated from a single concept and prompt start distribution. We sample 10 different sets (leading to 10 training set IDs) of 4 examples and generate all 24 possible permutations for each set of examples. Similarly to the behavior of GPT-3 (Zhao et al., 2021), there is a significant variation between example orderings, with a 10–40% gap between different permutations of the same set of examples.

E.6 Zero-shot is sometimes better than few-shot

On some hyperparameter settings of GINC, we find that zero-shot performance can be better than few-shot performance in our experiments. This finding mirrors the behavior of GPT-3 on some datasets (e.g., LAMBADA, HellaSwag, PhysicalQA, RACE-m, CoQA/SAT analogies for smaller

Prompt example length	Test Acc (200–300 chars)
5 examples	
Short (200–300 chars)	69.8
Long (500–600 chars)	70.7
10 examples	
Short, duplicated examples	69.6
Short, independent examples	71.4

Table 1: Accuracies for 5-shot in-context learning of GPT-3 on a filtered LAMBADA test set with short examples (200–300 characters). Even though there is distribution mismatch with the test set, having longer examples improves the accuracy, supporting theoretical intuitions. The first two rows use 5 training examples in the prompt, while the last two rows use 10 training examples to equalize the total length.

models (Brown et al., 2020)), where zero-shot performance is better than one-shot performance. This occurs especially when the transition probabilities in GINC are lower entropy (controlled via a temperature parameter). For this experiment, we consider GINC with transition matrix temperature parameter 0.01 (instead of 0.1), 12 concepts, and vocabulary size 100. Zero-shot completions in this case are more possible to predict with only the test input. Figure 13 (right) shows that in this case, the in-context accuracy is initially worse than zero-shot with a few examples, but can still later stabilize to a higher value. We hypothesize that this phenomenon occurs since the few-shot setting introduces the OOD prompt structure, thus initially decreasing the accuracy.

E.7 Experiment on GPT-3

We conduct an additional experiment which shows that longer examples improve in-context learning in GPT-3 on the LAMBADA (Paperno et al., 2016) completion task.

Data. In this experiment, we define a short version of the LAMBADA test dataset (LAMBADA test-short) which contains only test examples with up to 200–300 characters in length. We also define two "training" datasets from which to sample examples for the in-context prompts from. The short training dataset (LAMBADA train-short) contains examples from the training set that are 200–300 characters in length, which matches the distribution of test-short. The long training dataset (LAMBADA train-long) contains training examples that are 500–600 characters long. We cut the number of examples in the larger of the two training datasets so that the two training datasets are equally sized (47 examples). For each test example, we sample 5 random training examples (5-shot learning).

We also consider equalizing the total length of the prompts in two ways. First, we consider duplicating the 5 short examples (if the examples are [1,2,3,4,5], duplicating refers to [1,2,3,4,5,1,2,3,4,5]). This allows for equalizing the total length without increasing the number of examples. As a skyline comparison, we also consider sampling 10 independent short examples, which contains more input-output pairs for the task.

Result. Table 1 shows that when evaluating only on LAMBADA test-short, 5-shot in-context learning using LAMBADA train-long improves the test accuracy by almost 1% compared to LAMBADA train-short, despite the long/short distribution mismatch between train and test. This supports intuitions from our theory. In comparison, simply increasing the total prompt length by duplicating the short examples does not improve the accuracy. Intuitively, the longer examples have additional information that is not directly related to mapping between the input and output, but can be leveraged to improve in-context learning by helping the model infer the latent concept. Using 5 long examples (as opposed to 5 short examples) closes about 56% of the gap between using 5 short examples and 10 independent short examples despite not adding additional examples or task-related information.