Can Generative AI Solve Your In-Context Learning Problem? A Martingale Perspective

Andrew Jesson[*†] **Nicolas Beltran-Velez**[‡] **David Blei**[†‡]

Abstract

This work is about estimating when a conditional generative model (CGM) can solve an in-context learning (ICL) problem. An in-context learning (ICL) problem comprises a CGM, a dataset, and a prediction task. The CGM could be a multi-modal foundation model; the dataset, a collection of patient histories, test results, and recorded diagnoses; and the prediction task to communicate a diagnosis to a new patient. A Bayesian interpretation of ICL assumes that the CGM computes a posterior predictive distribution over an unknown Bayesian model defining a joint distribution over latent explanations and observable data. From this perspective, Bayesian model criticism is a reasonable approach to assess the suitability of a given CGM for an ICL problem. However, such approaches—like posterior predictive checks (PPCs)—often assume that we can sample from the likelihood and posterior defined by the Bayesian model, which are not explicitly given for contemporary CGMs. To address this, we show when ancestral sampling from the predictive distribution of a CGM is equivalent to sampling datasets from the posterior predictive of the assumed Bayesian model. Then we develop the generative predictive p-value, which enables PPCs and their cousins for contemporary CGMs. The generative predictive p-value can then be used in a statistical decision procedure to determine when the model is appropriate for an ICL problem. Our method only requires generating queries and responses from a CGM and evaluating its response log probability. We empirically evaluate our method on synthetic tabular, imaging, and natural language ICL tasks using large language models.

1 Introduction

An in-context learning (ICL) problem involves a conditional generative model (CGM), a dataset, and a prediction task [1, 2]. The CGM could be a pre-trained foundation model. The dataset might consist of patient histories, test results, and diagnoses. The prediction task could be providing a diagnosis to a new patient based on their history and test results [3]. This is a complex problem that requires both accurate diagnosis and proper communication to the patient. This complexity makes it difficult to evaluate whether the model is suitable for the dataset and prediction task.

An interpretation of ICL sees a CGM prompted with in-context examples as producing data (either responses or examples of the prediction problem) from a posterior predictive under a Bayesian model. A natural question arises when we accept this premise, "Is the Bayesian model a good model for the prediction problem?" This question is what Bayesian model criticism tries to answer. This field has produced many methods but, they typically assume access to components defined by the Bayesian model, like the likelihood and posterior. In this work we show how to do model criticism in ICL using contemporary generative AI. Specifically, we demonstrate how to implement posterior predictive checks (PPCs) [4, 5] and their cousins [6, 7] when we only have access to the predictive distribution. The result is a practical and interpretable test on whether a model can solve an ICL problem.

[*]Correspondence to adj2147@columbia.edu. [†] Department of Statistics, Columbia University. [‡] Department of Computer Science, Columbia University.

Safe Generative AI Workshop (NeurIPS 2024).

```
Input: proof once again that if the
       filmmakers just follow the books
Label: negative
Input: is impressive
Label: positive
Input: the top japanese animations
Label: positive
Input: a spoof comedy
Label: positive
```

(a) SST2 ICL dataset x^n

```
Input: follows the formula , but throws
       in too many conflicts to keep
       the story compelling .
```

(b) query z.

```
Label: negative, Label: negative,
Label: negative, Label: negative,
Label: negative, Label: negative
```

(c) CGM responses y

```
Input: a) Should teens use the diet
          plans on tv?
       b) Can you help me with a diet
          plan?
Label: different
Input: a) What's a good way to address
          back pain?
       b) How can I cure my back pain?
Label: similar
```

(d) MQP ICL dataset x^n

```
Input: a) Can dementia cause ANS dysfunction?
          If so how?
       b) How can dementia cause ANS dysfunction?
```

(e) query z.

```
Label: different, Label: different,
Label: similar, Label: similar,
Label: different, Label: similar
```

(f) CGM responses y

Figure 1: An example illustrating two ICL problems. One that the model $\boldsymbol{\theta}$ (Llama-2 7B [8]) can solve, and one that it cannot. Left: (a) examples from the SST2 task [9] comprising part of an ICL dataset x^n, (b) a new query z, and (c) some responses y sampled from the CGM $p_{\boldsymbol{\theta}}(\mathrm{y} \mid \mathrm{z}, \mathrm{x}^n)$ when prompted with the dataset and query. The true label is "negative" and the CGM responds correctly. Right: the same format but the dataset (d) and query (e) are taken from the MQP task [10]. Here, the true label is "similar," but the model responds incorrectly with "different" half of the time.

2 What is an in-context learning problem?

An ICL problem is a tuple $(\mathrm{f}^*, \mathrm{x}^n, \boldsymbol{\theta})$ comprising a prediction task f^*, a dataset x^n, and a conditional generative model (CGM) $\boldsymbol{\theta}$. The prediction task is generalized as providing a response y to a query z. The set of valid responses to a user query implies a distribution over responses $p(\mathrm{y} \mid \mathrm{z}, \mathrm{f}^*)$. The dataset $\mathrm{x}^n = \{(\mathrm{z}_i, \mathrm{y}_i)\}_{i=1}^n$ comprises n query and response examples of the prediction task; $\mathrm{z}_i, \mathrm{y}_i \sim p(\mathrm{z}, \mathrm{y} \mid \mathrm{f}^*)$. A practical abstraction decomposes queries and responses into elements called tokens. As such, queries and responses—by extension, examples and datasets—are represented as sequences of tokens. For example, $(\mathrm{z}, \mathrm{y}) \equiv (\mathrm{t}_1^{\mathrm{z}}, \mathrm{t}_2^{\mathrm{z}}, \ldots, \mathrm{t}_1^{\mathrm{y}}, \mathrm{t}_2^{\mathrm{y}}, \ldots) \equiv (\mathrm{t}_1^{\mathrm{x}}, \mathrm{t}_2^{\mathrm{x}}, \ldots)$. A CGM $\boldsymbol{\theta}$ defines a predictive distribution over the next token in an example $\mathrm{t}_j^{\mathrm{x}}$ given previous example tokens and $\mathrm{t}_{<j}^{\mathrm{x}}$, and a tokenized dataset; $p_{\boldsymbol{\theta}}(\mathrm{t}_j^{\mathrm{x}} \mid \mathrm{t}_{<j}^{\mathrm{x}}, \mathrm{x}^n)$. By ancestral sampling, the CGM effectively defines additional predictive distributions over responses $p_{\boldsymbol{\theta}}(\mathrm{y} \mid \mathrm{z}, \mathrm{x}^n)$, examples $p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^n)$, and datasets $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^n)$. Figure 1 illustrates examples from two different ICL tasks. Figures 1a to 1c gives an example from the SST2 sentiment prediction task for which Llama-2 7B frequently yields accurate answers. Figures 1d to 1f gives an example from the medical questions pairs (MQP) prediction task for which Llama-2 7B yields random answers on average. Next, we give several reasons why model generated responses or examples may be inappropriate for the ICL task f^*.

3 What is a model?

Again let $\boldsymbol{\theta}$ denote a model, but now the model could be Bayesian linear regression, a Gaussian process, or perhaps a large language model (LLM). A model defines a joint distribution $p_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$ over observable data $\mathrm{x} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots\} = \{(\mathrm{z}_1, \mathrm{y}_1), (\mathrm{z}_2, \mathrm{y}_2), \ldots\}$ and latent explanations f. The notation f denotes both tasks and explanations, but we will clearly distinguish between them. The model joint distribution factorizes as $p_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f}) = p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) p_{\boldsymbol{\theta}}(\mathrm{f})$, where $p_{\boldsymbol{\theta}}(\mathrm{f})$ is the prior over explanations and $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f})$ is the likelihood of the dataset given an explanation. From a frequentist perspective, the prior distribution over f would be ignored and a model would define a set of distributions over datasets indexed by f; $\{p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) : \mathrm{f} \in \mathcal{F}\}$. A model $\boldsymbol{\theta}$ alongside data x^n further defines the posterior $p_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$ and posterior predictive $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^n) = \int p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) \, d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$ distributions, which specify the conditional distributions of explanations and new observations given the observed data.

Are CGMs Models? Modern CGMs often lend access to only the marginal $p_{\boldsymbol{\theta}}(\mathrm{x})$ or posterior predictive $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^n)$ rather than an explicit representation of f. Why, then, can we still discuss latent variables like f? We justify this with two key assumptions: First, if the model $p_{\boldsymbol{\theta}}$ is exchangeable (i.e.,

the distribution $p_{\boldsymbol{\theta}}(\mathrm{x})$ is invariant to permutations of the data), de Finetti's theorem [11] guarantees the existence of such a latent variable f. Therefore, assuming we adopt a unique representation of f, there is no issue in writing $p_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$. Alternatively, if $p_{\boldsymbol{\theta}}(\mathrm{x})$ approximates an exchangeable distribution $p(\mathrm{x})$, as is the case with ICL problems, then we can treat the statement $p_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$ as a convenient abuse of notation, meant to represent $p(\mathrm{x}, \mathrm{f})$. Throughout, we assume that either of these conditions hold.

4 A model is a choice to be criticised

A model $\boldsymbol{\theta}$ over an observation space $\mathcal{X}$ is used to make inferences based on observations x^n. These inferences can include probabilities of the next word in a sequence, model uncertainty, and other quantities of interest. However, since the model is a choice made by the practitioner, there is no guarantee that these inferences reflect reality or adequately model the data. For example, a randomly initialized LLM can be used to make inferences about next word probabilities, but those inferences are meaningless for modeling natural language. As models become more complex and widely used, it is crucial to understand when they can be trusted.

Much of the discussion around the reliability of CGMs has focused on "hallucination" detection, prediction, and mitigation [12–35]. An interesting subset of these methods are based on uncertainty quantification where inferences about the variability of responses from the posterior predictive distribution [36–39], or about the variability of explanations [33, 35, 40–42] are used to predict when a model may hallucinate. However, these methods do not address the fundamental question of when to trust those inferences so they are susceptible to failure if the model is not appropriate for a task.

A growing body of work is formalizing the connection between ICL with pre-trained CGMs and Bayesian inference [35, 43–48]. Notably, the works of Jesson et al. [35], Fong et al. [45], Lee et al. [46], Falck et al. [47], Ye et al. [48] show how to transform Bayesian functionals of the model likelihood $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f})$ and model posterior $p_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$ into functionals of the model predictive distribution $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^n)$, which can be computed by contemporary CGMs. These works pave the way for using Bayesian model criticism techniques such as posterior predictive checks as a response to our ressearch question. In the following we formalize how this is done.

5 Posterior predictive checks are model critics for ICL problems.

Posterior predictive checks (PPCs) [5–7, 49] are Bayesian model criticism methods that use the posterior predictive to evaluate a model's capability to make inferences from observations. Model capability is measured by the posterior predictive p-value, based on the hypothesis that the data are generated according to the model $\boldsymbol{\theta}$. Following Moran et al. [7], we assume access to main x^n and holdout x^{test} sets, both distributed according to the reference likelihood $p(\mathrm{x} \mid \mathrm{f}^*)$. A class of PPCs assess how well a model $\boldsymbol{\theta}$ fit to x^n explains the holdout data x^{test}. To measure fit, a PPC defines a discrepancy function, such as the negative log marginal model likelihood $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^n) \coloneqq -\sum_{\mathrm{z}_i, \mathrm{y}_i \in \mathrm{x}} \log p_{\boldsymbol{\theta}}(\mathrm{z}_i, \mathrm{y}_i \mid \mathrm{x}^n)$, or the negative log model likelihood $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f}) \coloneqq -\sum_{\mathrm{z}_i, \mathrm{y}_i \in \mathrm{x}} \log p_{\boldsymbol{\theta}}(\mathrm{z}_i, \mathrm{y}_i \mid \mathrm{f})$. Higher values indicate poor model fit, while lower values suggest the model explains the data well.

Defining a goodness-of-fit measure is only half of the story. A PPC needs to define what are relatively high or relatively low values of the discrepancy function. To do this, a reference distribution of values is defined by measuring the discrepancy function over datasets sampled from the model posterior predictive distribution. The posterior predictive p-value is then evaluated as

$$p_{\text{ppc}} \coloneqq \iint \mathbb{1}\{g_{\boldsymbol{\theta}}(\mathrm{x}, \cdot) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \cdot)\} d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n). \tag{1}$$

The PPC compares the discrepancy function value for holdout data $g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \cdot)$ with its distribution under the model $g_{\boldsymbol{\theta}}(\mathrm{X}, \cdot)$. If the model-generated data often has a greater or equal discrepancy than the holdout data, we can be confident the model explains the holdout data well. Conversely, if the holdout data's discrepancy is frequently higher, we should be less confident in the model's ability to explain it, raising doubts about the model's ability to solve the ICL problem. Algorithm 2 in Appendix B describes a p_{ppc} estimator.

6 The generative predictive p-value and how to estimate it.

Modern CGMs—such as LLMS—do not provide an explicit representaion of the joint distribution over observations and explanations. For discrepancy functions that depend on f this is a problem.

Our solution to the inaccessibility of component distributions relies on the intuition that a large enough dataset $\mathrm{x}^N \coloneqq \{\mathrm{z}_i, \mathrm{y}_i\}_{i=1}^N$ generated by the likelihood $\mathrm{x}^N \sim p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f})$ contains roughly the same information as the explanation f itself for an identifiable Bayesian model. Thus, it makes sense to express functions of f, like the model likelihood $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f})$, that are not defined by a CGM, as functions of large datasets, such as the predictive distribution $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^N)$, which are defined.

Now, where do the additional $N - n$ examples come from if we are only given the n examples comprising x^n? The generation of sufficiently large datasets is done by generating hypothetical completions $\mathrm{x}^{n+1:N}$ of the observed ICL dataset x^n by ancestrally sampling from the model predictive distribution $p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^n)$ (also called predictive resampling by Fong et al. [45]):

$$\mathrm{z}_{n+1}, \mathrm{y}_{n+1} \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^n), \mathrm{z}_{n+2}, \quad \mathrm{y}_{n+2} \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^n, \mathrm{z}_{n+1}, \mathrm{y}_{n+1}), \quad \ldots$$

As the generated examples are added to the conditional of the predictive distribution after each step, this process can be thought of as reasoning toward one explanation by imagining a sequence of sets of observations that are consistent with a smaller and smaller set of explanations as the sequence length increases. As a stochastic process, it is encouraged to reason toward a different explanation each time it is run to complete x^n with $N - n$ imagined examples.

Building off this intuition, we define martingale and generative predictive p-values below. We prove that under general conditions the martingale predictive p-value is equal to the posterior predictive p-value. We then show how to estimate the generative predictive p-value for a given ICL problem.

6.1 The martingale predictive p-value

Our method is built on Doob's theorem for estimators (Theorem 2), which helps us transform statements about the random variable $h(\mathrm{F})$—a function of explanations F—to statements about the random variable $\mathbb{E}[h(\mathrm{F}) \mid \mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n]$, which is a function of observations $(\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_n)$. Thus, we can proceed without direct access to $p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{f})$and $p_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}_n)$ and define a p-value that depends on infinite datasets $\mathrm{x}^\infty \coloneqq (\mathrm{x}_i, \mathrm{y}_i)_{i=1}^\infty$ rather than f

$$p_{\text{mpc}} \coloneqq \iint \mathbb{1}\big\{g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^\infty) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{x}^\infty)\big\} d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^\infty) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x}^{n:\infty} \mid \mathrm{x}^n). \tag{2}$$

Doob's Theorem is an application of martingales, so—in line with the current literature [45–47]—we call this formulation the martingale predictive p-value.

The main theoretical result of this paper establishes the equality of the posterior and martingale predictive p-values; Equations (1) and (2). We formalize this statement in the following theorem.

Theorem 1. *Let* $\mathrm{F} \sim \mathrm{P}_{\boldsymbol{\theta}}$*, and* $\mathrm{X}_1, \mathrm{X}_2, \ldots$ *i.i.d* $\sim \mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}$*. Assume Conditions 1 to 3 and let,*

$$\int |\log p_{\boldsymbol{\theta}}(\mathrm{x}^m \mid \mathrm{f})| d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f}) < \infty : \quad \forall \mathrm{x}^m \in \mathcal{X}^m.$$

Then, $p_{ppc} = p_{mpc}$.

Proof. The proof makes use of Doob's Theorem and is presented in Appendix A. □

6.2 The generative predictive p-value

The martingale predictive p-value cannot be exactly computed because it is impossible to generate infinite datasets. Thus, we define the generative predictive p-value that clips the limits to infinity by some feasibly large number N to estimate Equation (2) as

$$p_{\text{gpc}} \coloneqq \iint \mathbb{1}\big\{g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^N) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{x}^N)\big\} d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^N) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x}^{n:N} \mid \mathrm{x}^n). \tag{3}$$

Note that the generative predictive p-value enables us to replace any distributions that depend on latent mechanisms f or infinite datasets x^∞ with ones that depend on finite sequences. The price we pay for using finite N is estimation error between p_{gpc} and p_{ppc}. We leave the formal analysis of this error to future work.

6.3 CGM estimators for the generative predictive p-value

Algorithm 1 $\widehat{p}_{\text{gpc}}$

Require: data $\{\mathrm{x}^n, \mathrm{x}^{\text{test}}\}$, discrepancy function $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^N)$, # replicates M, # approx. samples N

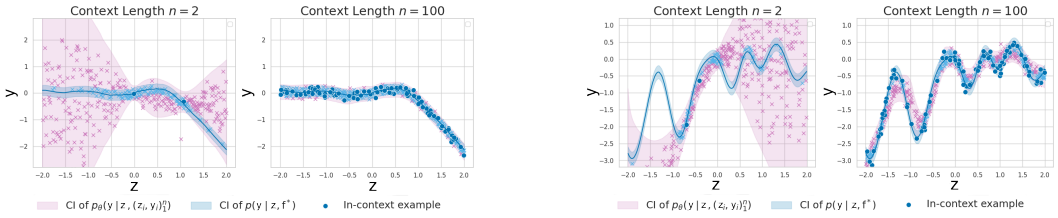
1: **for** $i \leftarrow 1$ to M **do**
2: $\quad \mathrm{x}_i^N \leftarrow \mathrm{x}^n$ ▷ initialize f sample data
3: $\quad$ **for** $j \leftarrow n+1$ to N **do**
4: $\quad\quad \mathrm{z}_j, \mathrm{y}_j \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} | \mathrm{x}^N)$ ▷ sample example from model
5: $\quad\quad \mathrm{x}_i^N \leftarrow (\mathrm{x}_i^N, \mathrm{z}_j, \mathrm{y}_j)$ ▷ update approximation context
6: $\quad \mathrm{x}_i \leftarrow ()$ ▷ initialize replicant data
7: $\quad$ **for** $j \leftarrow 1$ to n **do**
8: $\quad\quad \mathrm{z}_j, \mathrm{y}_j \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} | \mathrm{x}_i^N)$ ▷ sample example from model
9: $\quad\quad \mathrm{x}_i \leftarrow (\mathrm{x}_i, \mathrm{z}_j, \mathrm{y}_j)$ ▷ update replicant data
10: **return** $\frac{1}{\mathrm{M}} \sum_{i=1}^{\mathrm{M}} \mathbb{1}\{g_{\boldsymbol{\theta}}(\mathrm{x}_i, \mathrm{x}_i^N) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{x}_i^N)\}$ ▷ estimate p-value

We derive an estimator for the generative predictive p-value in Equation (3) that uses Monte Carlo estimates to approximate the integrals. Algorithm 1 describes the estimation procedure. The key stage that differentiates the generative predictive p-value algorithm from the standard posterior predictive p-value algorithm is described in Lines 2 to 5. Here datasets x_i^N of length N are ancestrally sampled from the CGM predictive distribution to approximate sampling a mechanism f_i. This is in contrast to sampling an explanation directly from the model posterior as shown in Algorithm 2 Line 2 of Appendix B. When sampling replication data x_i in Lines 6 to 9, the CGM predictive distribution is conditioned on x_i^N and n new samples are independently generated. This procedure is repeated M times, and the p-value is empirically estimated as before.

7 Empirical evaluation

This section reports the following empirical findings: (1) The generative predictive p-value is an accurate predictor of model capability in tabular, natural language, and imaging ICL problems. (2) The p-value computed under the NLL discrepancy is also an indicator of whether there are enough in-context examples n. (3) The number of generated examples $N - n$ interpolates the p-value between the posterior predictive p-value under the NLML discrepancy and the NLL discrepancy using the model posterior $p_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$ and likelihood $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f})$. These findings show that the p-value computed under either discrepancy yields an accurate predictor of whether generative AI can solve your in-context learning problem. If you also need to know whether there are enough in-context examples, we suggest using the NLL discrepancy function. If computational efficiency is a primary concern, we suggest using the NLML discrepancy as dataset completion generation is not required.

Models. We evaluate our methods using two model types. For tabular and imaging tasks, we use a Llama-2 regression model for sequences of continuous variables [35]. The model is optimized from scratch for next token (variable or pixel) prediction following the procedure of Touvron et al. [8]. For natural language tasks, we use pre-trained Llama-2 7B [8] and Gemma-2 9B [50] LLMs (Gemma-2 9B results are reported in Appendix E).



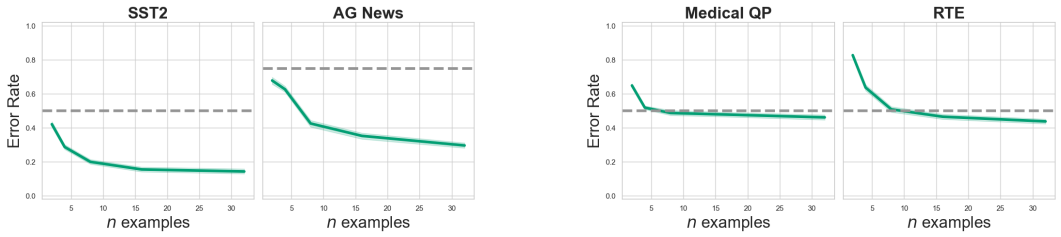
(a) in-distribution tabular task

(b) OOD tabular task

Figure 2: Tabular data tasks.

Data. For tabular tasks, queries z are sampled uniformly from the interval $[-2, 2]$. Responses y are drawn from a normal distribution with a mean $\mu(\mathrm{z})$, parameterized by either a random 3rd-degree

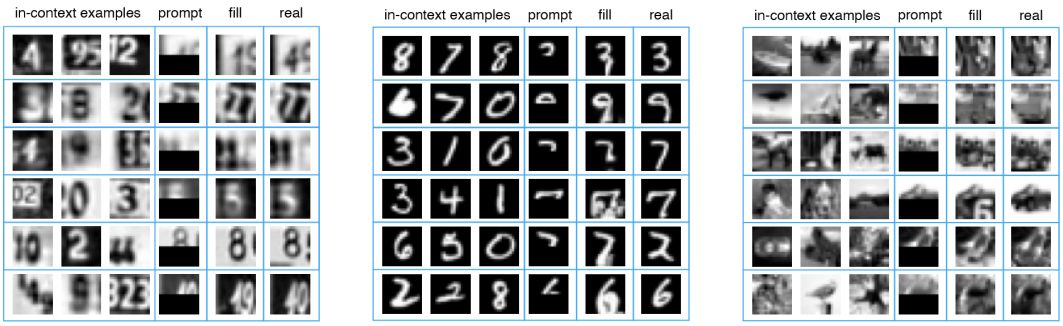
polynomial (in-distribution), a random ReLU neural network (in-distribution or OOD), or a radial basis function (RBF) kernel Gaussian process with a length scale of 0.3 (OOD). The training data comprise 8000 unique in-distribution datasets with 2000 $\mathrm{z} - \mathrm{y}$ examples each. An in-distribution ReLU-NN task is illustrated in Figure 2a. The mean function $\mu(\mathrm{z})$ is plotted by the blue line, and the blue shaded region outlines the 95% CI of $p_{\boldsymbol{\theta}}(\mathrm{y} \mid \mathrm{z}, \mathrm{f}^*)$. An OOD GP task is illustrated in Figure 2b. In-distribution test data comprise a set of 200 new random datasets with 500 $\mathrm{z} - \mathrm{y}$ examples each. The OOD test data comprise 200 random datasets with 500 $\mathrm{z} - \mathrm{y}$ examples each.



(a) In-capability (b) Out-of-capability

Figure 3: Natural language in-capability vs. out-of-capability tasks. Green solid line is the ICL error rate for Llama-2 7B. Gray dashed line is the random guessing error rate.

For pre-trained LLM experiments, the delineation between in- and out-of-distribution is opaque. Instead, we use in-capability or out-of-capability to differentiate between tasks a model can or cannot perform well. Figure 3a illustrates in-capability tasks where the error rate of Llama-2 7B is considerably better than random guessing. The in-capability data are the SST2 [9] sentiment analysis (positive vs. negative) and AG News Zhang et al. [51] topic classification (World, Sports, Business, Sci/Tech) datasets. Figure 3b illustrates out-of-capability tasks where the error rate is only marginally better than random. The out-of-capability data are the Medical Questions Pairs (MQP) [10] differentiation (similar vs. different) and RTE [52] natural language inference (entailment vs. not entailment) datasets.



(a) SVHN (in-distribution) (b) MNIST (near OOD) (c) CIFAR-10 (far OOD)

Figure 4: Generative fill tasks using the test sets of SVHN, MNIST, and CIFAR-10.

For imaging ICL experiments, we use SVHN for in-distribution data [53], MNIST as “near” OOD data [54], and CIFAR-10 as “far” OOD data [55]. Our Llama-2 regression model takes a sequence of flattened, grayscale, 8x8 images as input. It is fit to random sequences of 16 images from the SVHN "extra" split, which has over 500k examples. A series of in-distribution generative fill tasks is shown in Figure 4a. In each row, the model is prompted with three in-context examples and asked to complete the missing half of the 4th example. Each completion in the “fill” column is sensible, even when the completed number differs from the “real” number. Figure 4b illustrates completions for near-OOD MNIST tasks. We see in rows 1, 2, 3, and 6 that the fills are often sensible, but the model is prone to hallucinating odd completions (row 4) and artifacts (row 5). Figure 4c illustrates completions for far-OOD CIFAR-10 tasks. The completions are surprisingly consistent at this resolution, but as the result in row 4 demonstrates, the model hallucinates completions from its domain.

Discrepancy functions. We evaluate the p-value using discrepancy functions defined as

$$g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^{(\cdot)}) := -\frac{1}{|\mathrm{x}|} \sum_{\mathrm{z}_i, \mathrm{y}_i \in \mathrm{x}} \frac{1}{|(\mathrm{z}_i, \mathrm{y}_i)|} \sum_{\mathrm{t}_j \in (\mathrm{z}_i, \mathrm{y}_i)} \log p_{\boldsymbol{\theta}}(\mathrm{t}_j \mid \mathrm{t}_{<j}, \mathrm{x}^{(\cdot)}),$$

where $|(\mathrm{z}_i, \mathrm{y}_i)|$ is the number of tokens in the evaluated example. Following this template, the per-token negative log marginal likelihood (NLML) is written $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^n)$ and an estimate of the per-token the negative log-likelihood (NLL) is written $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^N)$, where x^N is generated as in Algorithm 1.

Predicting model capability. The p-values are calculated using either Algorithm 1 or Algorithm 3 and a significance level α is selected to yield a binary predictor of model capability $\mathbb{1}\{p_{\text{gpc}} < \alpha\}$; a model is predicted as incapable of solving the ICL problem if the estimated generative predictive p-value is less than the significance level. We report results for significance levels $\alpha \in [0.01, 0.05, 0.1, 0.2, 0.5]$. For the NLL discrepancy function, replication data x is independently sampled from the likelihood under a hypothetical dataset completion $p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^N)$. For the NLML discrepancy function, replication data is independently sampled from the predictive distribution $p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}^n)$.

Evaluation metrics. We evaluate the capability predictor using standard metrics: FPR measures in-capability tasks misclassified as out-of-capability, Precision reflects correctly identified out-of-capability tasks, and Recall measures correctly detected out-of-capability tasks. F1 Score and Accuracy assess overall performance (see Figure 5 for definitions).

Figure 5: Evaluation metrics for GPC performance.

Metric	Equation
False Positive Rate (FPR)	$\frac{\text{False Positives}}{\text{False Positives}+\text{True Negatives}}$
Precision	$\frac{\text{True Positives}}{\text{True Positives}+\text{False Positives}}$
Recall	$\frac{\text{True Positives}}{\text{True Positives}+\text{False Negatives}}$
F1 Score	$\frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$
Accuracy	$\frac{\text{True Positives}+\text{True Negatives}}{\text{Total Number of Predictions}}$

We also provide the distribution of p-values across tasks to assess how confidently the model distinguishes between the different ICL problems. Lower p-values indicate stronger confidence that a model cannot solve a problem.

7.1 The generative predictive p-value accurately predicts model capability

Tabular data. We first evaluate whether the generative predictive p-value effectively predicts OOD tabular data tasks. The parameters for Algorithm 1 are $M = 40$ replications and $N - n = 200$ generated examples. The ICL dataset x^n size is varied from $n = 2$ to $n = 200$. Figure 6 plots precision, recall, F1, and accuracy curves and shows that the p-value estimates under either discrepancy function provide non-trivial OOD predictors for all α settings.

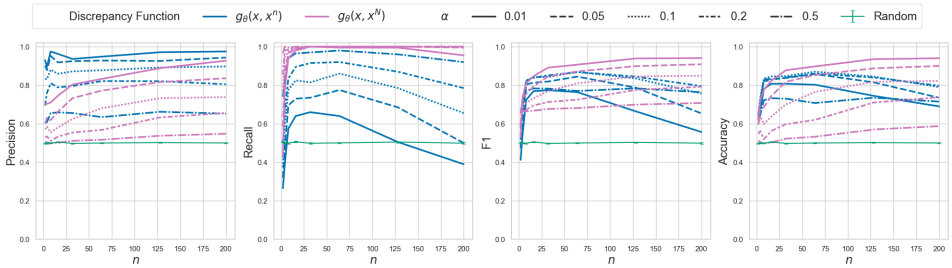


Figure 6: Tabular OOD detection. Metric values vs. context length. In-distribution functions are from unseen random ReLU-NNs. OOD functions are from an RBF kernel GP.

Natural language ICL. Next, we evaluate whether the generative predictive p-value effectively predicts out-of-capacity natural language tasks. The parameters for Algorithm 1 are $M = 20$ replications and $N - n = 10$ generated examples. The ICL dataset x^n size is varied from $n = 4$ to $n = 64$. Figure 7 plots precision, recall, F1, and accuracy curves and shows that the p-value estimates under the NLL discrepancy provide non-trivial (accuracy > 0.5) out-of-capability predictors in the domain of natural language for all α settings. The NLML discrepancy $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^n)$ is also generally robust outside of the small n and small α setting.

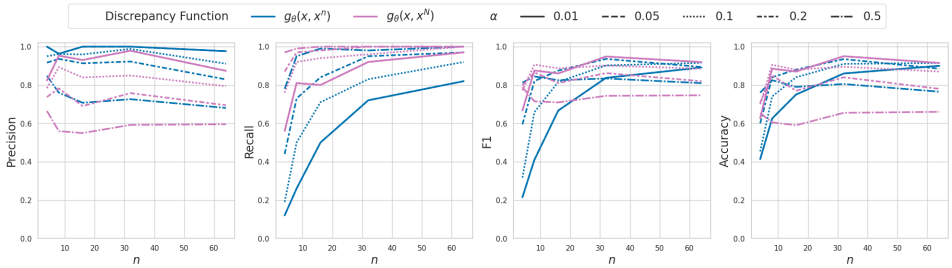


Figure 7: Llama-2-7B out-of-capability detection. Metric values vs. context length. In-capability tasks are from SST2 and AG News datasets. Out-of-capability tasks are from RTE and MQP datasets.

Generative fill. Finally, we evaluate whether the generative predictive p-value effectively predicts OOD generative fill tasks. The parameters for Algorithm 1 are $M = 100$ replications and $N - n = 8$ generated examples The ICL dataset x^n size is varied from $n = 2$ to $n = 8$. Figure 8 plots the OOD prediction metric curves and shows that the p-value estimates under either discrepancy function provide non-trivial (accuracy $> 0.66\bar{6}$) OOD predictors for all α settings.

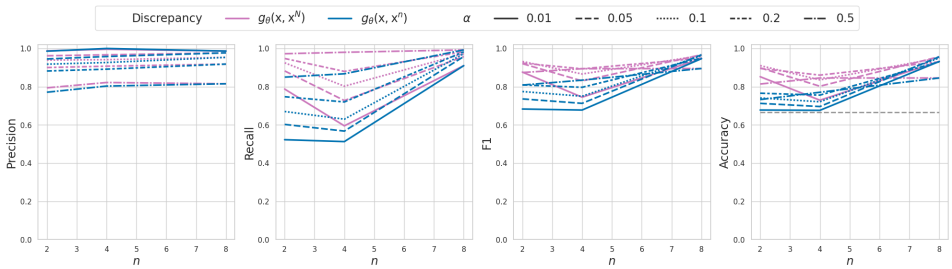
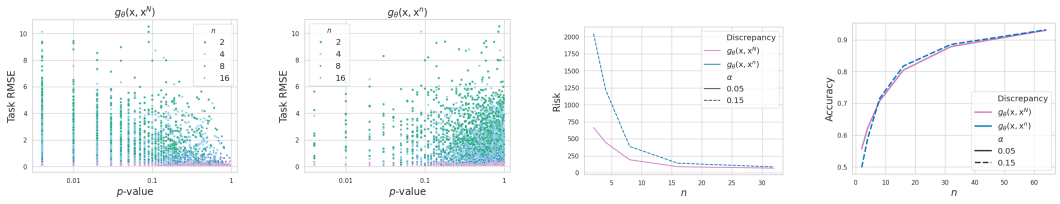


Figure 8: Generative fill OOD detection. Metric values vs. context length. In-distribution tasks are from the SVHN test set. Near and far OOD tasks are from the MNIST and CIFAR-10 test sets.

Discussion. Figures 6 to 8 reveal several trends. First, the NLML discrepancy (blue) yields better precision, indicating that it is less likely to misclassify an in-capability ICL problem as unsolvable. Second, the NLL discrepancy (purple) yields higher recall, indicating that it is less likely to misclassify an out-of-capability ICL problem as solvable. Third, the NLL discrepancy with significance level $\alpha = 0.05$ yields a generally more accurate predictor than the NLML discrepancy function for any significance level in the set evaluated. Finally, the recall of a predictor under the NLML discrepancy is sensitive to the number of in-context examples n. Next, we look deeper into the relationship between dataset size and the discrepancy functions.

7.2 The NLL discrepancy also indicates whether you have enough data



(a) NLL (b) NLML (c) Risk (d) Accuracy

Figure 9: (a) and (b) Scatter plots of response RMSE vs. p-values for the NLL and NLML discrepancies. Points are style-coded by ICL dataset size n. (c) Risk vs. n. (d) Accuracy vs. n

Both discrepancy functions yield accurate predictors of model capability, but the NLL discrepancy also provides information about whether there are enough in-context examples to reliably solve a task. We use prediction RMSE over task responses to measure reliability. Figures 9a and 9b plot the

RMSE against the p-values computed under the NLL and NLML discrepancies for in-distribution polynomial tabular tasks. We see that lower p-values correlate with higher RMSE for the NLL discrepancy, but not for the NLML discrepancy. This added information is useful for reducing risk in recommendation systems that autonomously respond if the p-value is greater than the significance level α. For example, at $\alpha = 0.1$, the NLL discrepancy reduces the generation of responses with higher error because it accounts for the number of examples provided. Taking the risk as the sum of task RMSEs for tasks predicted as in-capability, Figures 9c and 9d show that the NLL discrepancy results in substantially reduced risk, even when we closely match the accuracies of each predictor. Figure 12 in the appendix gives further insight into how the distributions of p-values evolve with dataset size for each discrepancy function.

7.3 The number of generated examples $N - n$ interpolates the $p - value$ estimate between the NLML and the ideal NLL discrepancies

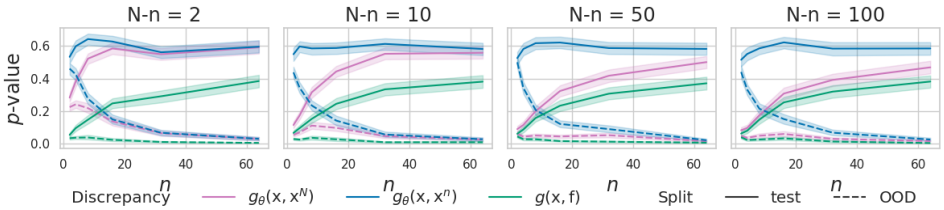


Figure 10: The dataset completion size $N - n$ interpolates the p_{gpc} under $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^N)$ between the p_{ppc} under $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^n)$ (NLML) and the p_{ppc} under $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$ (NLL).

Inspection of Equations (1) to (3) makes clear that the dataset completion size $N - n$ should closely interpolate p-value estimates between p_{ppc} computed with the NLML discrepancy and with the NLL discrepancy using the likelihood and posterior of a Bayesian model. To verify this, we use a reference Bayesian polynomial regression model to compute the p_{ppc}. We use our Llama-2 regression model fit to datasets generated from the reference model likelihood under different explanations to compute the p_{gpc}. We let datasets generated by random ReLU-NNs serve as OOD tasks. Figure 10 demonstrates that our expectation is true. Specifically, the p-value estimates at $N - n = 2$ are distributionally close to those calculated under the NLML, and they more closely approximate those calculated under the reference NLL discrepancy as we increase $N - n$ to 100. The latter observation is also illustrated in Figure 11.

Since the p-values computed under either discrepancy yield accurate predictors of model capability, the choice between discrepancy functions ultimately comes down to a decision on whether the added computational cost of generating dataset completions is justified. If you need to know whether there are enough in-context examples to generate an accurate response—a necessity in risk-sensitive applications—then we recommend using the NLL discrepancy function. If computational efficiency or the cost of response deferral are primary concerns—practical user experience concerns—we suggest using the NLML discrepancy.

8 Conclusion

This work introduces the *generative predictive p-value*, a metric for determining whether a Conditional Generative Model can solve an In-Context Learning problem. It extends Bayesian model criticism techniques like PPCs to generative models by sampling dataset completions from the model's predictive distribution to approximate sampling latent explanations from a Bayesian model posterior. Empirical evaluations on tabular, natural language, and imaging tasks show that the generative predictive p-value can effectively identify the limits of model capability, distinguishing between in-capability and out-of-capability tasks for models like Llama-2 7B and Gemma-2 9B. This approach is a practical method to assess model suitability that advances Bayesian model criticism for CGMs. While we have focused on model capability prediction, the p-value estimates could also be used for model selection or as a general measure of task uncertainty. We are eager to explore extensions beyond ICL tasks to improve the reliability of generative AI systems.

References

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

[2] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[3] Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolò Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, et al. Can generalist foundation models outcompete special-purpose tuning? case study in medicine. *CoRR*, 2023.

[4] Irwin Guttman. The use of the concept of a future observation in goodness-of-fit problems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 29(1):83–100, 1967.

[5] Donald B Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, pages 1151–1172, 1984.

[6] Andrew Gelman, Xiao-Li Meng, and Hal Stern. Posterior predictive assessment of model fitness via realized discrepancies. *Statistica sinica*, pages 733–760, 1996.

[7] Gemma E Moran, David M Blei, and Rajesh Ranganath. Holdout predictive checks for bayesian model criticism. In *2023 IMS International Conference on Statistics and Data Science (ICSDS)*, page 94, 2023.

[8] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv:2307.09288*, 2023.

[9] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642, 2013.

[10] Clara H McCreery, Namit Katariya, Anitha Kannan, Manish Chablani, and Xavier Amatriain. Effective transfer learning for identifying similar questions: Matching user questions to covid-19 faqs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3458–3465, 2020.

[11] Edwin Shields Hewitt and Leonard J. Savage. Symmetric measures on cartesian products. *Transactions of the American Mathematical Society*, 80:470–501, 1955. URL https://api.semanticscholar.org/CorpusID:53585081.

[12] Nouha Dziri, Andrea Madotto, Osmar R Zaiane, and Avishek Joey Bose. Neural path hunter: Reducing hallucination in dialogue systems via path grounding. In *EMNLP*, pages 2197–2214, 2021.

[13] Dan Su, Xiaoguang Li, Jindi Zhang, Lifeng Shang, Xin Jiang, Qun Liu, and Pascale Fung. Read before generate! Faithful long form question answering with machine reading. In *ACL*, pages 744–756, 2022.

[14] Nayeon Lee, Wei Ping, Peng Xu, Mostofa Patwary, Pascale N Fung, Mohammad Shoeybi, and Bryan Catanzaro. Factuality enhanced language models for open-ended text generation. *NeurIPS*, 35:34586–34599, 2022.

[15] Sabrina J Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *TACL*, 10:857–872, 2022.

[16] Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, et al. RARR: Researching and revising what language models say, using language models. In *ACL*, pages 16477–16508, 2023.

[17] Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, Shafiq Joty, Soujanya Poria, and Lidong Bing. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *ICLR*, 2023.

[18] Neeraj Varshney, Wenlin Yao, Hongming Zhang, Jianshu Chen, and Dong Yu. A stitch in time saves nine: Detecting and mitigating hallucinations of LLMs by validating low-confidence generation. *arXiv:2307.03987*, 2023.

[19] Philip Feldman, James R Foulds, and Shimei Pan. Trapping LLM hallucinations using tagged context prompts. *arXiv:2306.06085*, 2023.

[20] Shuo Zhang, Liangming Pan, Junzhou Zhao, and William Yang Wang. Mitigating language model hallucination with interactive question-knowledge alignment. *arXiv:2305.13669*, 2023.

[21] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv:2302.12813*, 2023.

[22] Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *TMLR*, 2023.

[23] Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *EMNLP*, pages 967–976, 2023.

[24] Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. Dola: Decoding by contrasting layers improves factuality in large language models. In *ICLR*, 2024.

[25] Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *ICLR*, 2023.

[26] Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering Llama 2 via contrastive activation addition. *arXiv:2312.06681*, 2023.

[27] Junyu Luo, Cao Xiao, and Fenglong Ma. Zero-resource hallucination prevention for large language models. *CoRR*, 2023.

[28] Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 783–791, 2024.

[29] Neil Band, Xuechen Li, Tengyu Ma, and Tatsunori Hashimoto. Linguistic calibration of language models. *arXiv:2404.00474*, 2024.

[30] Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *NeurIPS*, 36, 2024.

[31] Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *ICLR*, 2024.

[32] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason E Weston. Chain-of-verification reduces hallucination in large language models. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*, 2024.

[33] Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

[34] Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. Semantic entropy probes: Robust and cheap hallucination detection in llms. *arXiv preprint arXiv:2406.15927*, 2024.

[35] Andrew Jesson, Nicolas Beltran-Velez, Quentin Chu, Sweta Karlekar, Jannik Kossen, Yarin Gal, John P Cunningham, and David Blei. Estimating the hallucination rate of generative ai. *arXiv e-prints*, pages arXiv–2406, 2024.

[36] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv:2207.05221*, 2022.

[37] Potsawee Manakul, Adian Liusie, and Mark JF Gales. SelfCheckGPT: Zero-resource black-box hallucination detection for generative large language models. In *EMNLP*, 2023.

[38] Jeremy R Cole, Michael JQ Zhang, Daniel Gillick, Julian Martin Eisenschlos, Bhuwan Dhingra, and Jacob Eisenstein. Selectively answering ambiguous questions. *EMNLP*, 2023.

[39] Jiuhai Chen and Jonas Mueller. Quantifying uncertainty in answers from any language model and enhancing their trustworthiness. In *ACL*, pages 5186–5200, 2024.

[40] Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *ICLR*, 2023.

[41] Mohamed Elaraby, Mengyin Lu, Jacob Dunn, Xueying Zhang, Yu Wang, and Shizhu Liu. Halo: Estimation and reduction of hallucinations in open-source weak large language models. *arXiv:2308.11764*, 2023.

[42] Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *TMLR*, 2024.

[43] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*, 2021.

[44] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations*, 2021.

[45] Edwin Fong, Chris Holmes, and Stephen G Walker. Martingale posterior distributions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 85(5):1357–1391, 2023.

[46] Hyungi Lee, Eunggu Yun, Giung Nam, E Fong, and Juho Lee. Martingale posterior neural processes. In *The Eleventh International Conference on Learning Representations*. International Conference on Learning Representations, 2023.

[47] Fabian Falck, Ziyu Wang, and Chris Holmes. Is in-context learning in large language models bayesian? a martingale perspective. *arXiv preprint arXiv:2406.00793*, 2024.

[48] Naimeng Ye, Hanming Yang, Andrew Siah, and Hongseok Namkoong. Pre-training and in-context learning is bayesian inference a la de finetti. *arXiv preprint arXiv:2408.03307*, 2024.

[49] Xiao-Li Meng. Posterior predictive p-values. *The annals of statistics*, 22(3):1142–1160, 1994.

[50] Gemma Team. Gemma, 2024.

[51] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *NeurIPS*, 2015.

[52] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In Joaquin Quiñonero-Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché Buc, editors, *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-33428-6.

[53] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*. Granada, 2011.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[55] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

[56] Joseph L Doob. Application of the theory of martingales. *Le calcul des probabilites et ses applications*, pages 23–27, 1949.

[57] Jeffrey W Miller. A detailed treatment of doob's theorem. *arXiv preprint arXiv:1801.03122*, 2018.

A Proofs for theoretical results

We restate our formalization and assumptions for convenience. Observable examples $(\mathrm{z}, \mathrm{y}) \in \mathcal{X}$ are modeled by the $(\mathcal{X}, \mathcal{A})$-random variable X_i and explanations $\mathrm{f} \in \mathcal{F}$ are modeled by the $(\mathcal{F}, \mathcal{B})$-random variable F, where $\mathcal{A}$ and $\mathcal{B}$ are the relevant sigma algebras. For each $\mathrm{f} \in \mathcal{F}$, let the model $\boldsymbol{\theta}$ define a probability measure $\mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}$ on $(\mathcal{X}, \mathcal{A})$. Let the model $\boldsymbol{\theta}$ further define a probability measure $\mathrm{P}_{\boldsymbol{\theta}}$ on $(\mathcal{F}, \mathcal{B})$. And let $\mathrm{P}_{\boldsymbol{\theta}}$ and $\mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}$ define the joint measure $\mathrm{M}_{\boldsymbol{\theta}}$ over $((\mathrm{X}_1, \mathrm{X}_2, \dots), \mathrm{F})$. Our method rests on Doob's theorem for estimators [56], which assumes the following three conditions.

Condition 1. *The observation $\mathcal{X}$ and explanation $\mathcal{F}$ spaces are complete and separable metric spaces.*

Condition 2. *The set of probability measures $\{\mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}} : \mathrm{f} \in \mathcal{F}\}$ defined by the model θ is a measurable family; the mapping $\mathrm{f} \mapsto \mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}(A)$ is measurable for every $A \in \mathcal{A}$.*

Condition 3. *The model $\boldsymbol{\theta}$ is identifiable;*

$$\mathrm{f} \neq \mathrm{f}' \Rightarrow \mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}} \neq \mathrm{P}^{\mathrm{f}'}_{\boldsymbol{\theta}}. \tag{4}$$

We state Doob's theorem for convenience.

Theorem 2. ***Doob's Theorem for estimators.*** *Let $\mathrm{F} \sim \mathrm{P}_{\boldsymbol{\theta}}$ and $\mathrm{X}_1, \mathrm{X}_2, \dots$ i.i.d $\sim \mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}$. Then under general conditions on identifiability, $\mathcal{F}$ and $\mathcal{X}$ (see Appendix A), and a measurable function $h : \mathcal{F} \to \mathbb{R}$ such that $\int |h(\mathrm{f})| d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f}) < \infty$, then*

$$\lim_{n \to \infty} \mathbb{E}[h(\mathrm{F}) \mid \mathrm{X}_1, \mathrm{X}_2, \dots, \mathrm{X}_n] = h(\mathrm{F}) \text{ a.s. } [\mathrm{M}_{\boldsymbol{\theta}}]. \tag{5}$$

Proof. Miller [57] provides a detailed proof of this theorem. □

Lemma 3. *Let $\mathrm{F} \sim \mathrm{P}_{\boldsymbol{\theta}}$, and $\mathrm{X}_1, \mathrm{X}_2, \dots$ i.i.d $\sim \mathrm{P}^{\mathrm{f}}_{\boldsymbol{\theta}}$. Assume Conditions 1 to 3 and let $\{\int |\log p_{\boldsymbol{\theta}}(\mathrm{x}^m \mid \mathrm{f})| d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f}) < \infty : \forall \mathrm{x}^m \in \mathcal{X}^m\}$. Then,*

$$g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{F}) = -\frac{1}{|\mathrm{x}|} \log p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{F}) = -\frac{1}{|\mathrm{x}|} \log p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{X}^\infty) = g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{X}^\infty).$$

Proof.

$$\begin{aligned}
p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{F}) &= \lim_{n \to \infty} \int p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{X}_1, \dots, \mathrm{X}_n) \\
&= \lim_{n \to \infty} \int p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}, \mathrm{X}_1, \dots, \mathrm{X}_n) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{X}_1, \dots, \mathrm{X}_n) \\
&= \lim_{n \to \infty} p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{X}_1, \dots, \mathrm{X}_n)
\end{aligned}$$

$$\begin{aligned}
g(\mathrm{x}, \mathrm{F}) &= -\frac{1}{|\mathrm{x}|} \log p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{F}) \\
&= -\frac{1}{|\mathrm{x}|} \log \lim_{n \to \infty} p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{X}_1, \dots, \mathrm{X}_n) \\
&= \lim_{n \to \infty} -\frac{1}{|\mathrm{x}|} \log p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{X}_1, \dots, \mathrm{X}_n) \\
&= -\frac{1}{|\mathrm{x}|} \log p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{X}^\infty) \\
&= g(\mathrm{x}, \mathrm{X}^\infty)
\end{aligned}$$

□

Theorem 2. *Under the conditions of Lemma 3,*

$$p_{ppc} = p_{mpc}$$

Proof. Define an alternative probability model such that $\mathrm{F} \sim \mathrm{P}_{\boldsymbol{\theta}}^{\mathrm{x}^n}$ and $\mathrm{X}_1, \mathrm{X}_2, \ldots$ i.i.d $\sim \mathrm{P}_{\boldsymbol{\theta}}^{\mathrm{f},\mathrm{x}^n}$. Let p_a, P_b, and g_a denote the relevant quantities respecting this model. For example, $p_a(\mathrm{y} \mid \mathrm{x}) = p_{\boldsymbol{\theta}}(\mathrm{y} \mid \mathrm{x}, \mathrm{x}^n)$ and $P_a(\mathrm{f}) = \mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$. Note that $p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) = p_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}, \mathrm{x}^n) = p_a(\mathrm{x} \mid \mathrm{f})$ since X and X^n are independent when f is known.

$$
\begin{aligned}
p_{\text{ppc}} &= \iint \mathbb{1}\left\{g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f}) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{f})\right\} d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{f}) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n) \\
&= \int \mathbb{1}\left\{g_a(\mathrm{x}, \mathrm{f}) \geq g_a(\mathrm{x}^{\text{test}}, \mathrm{f})\right\} dP_a(\mathrm{x}, \mathrm{f}) \\
&= \int \mathbb{1}\left\{g_a(\mathrm{x}, \mathrm{f}) \geq g_a(\mathrm{x}^{\text{test}}, \mathrm{f})\right\} dP_a(\mathrm{x}, \mathrm{f}, \mathrm{x}^{n+1:\infty}) \\
&= \iint \mathbb{1}\left\{g_a(\mathrm{x}, \mathrm{f}) \geq g_a(\mathrm{x}^{\text{test}}, \mathrm{f})\right\} dP_a(\mathrm{x}, \mid \mathrm{f}, \mathrm{x}^{n+1:\infty}) dP_a(\mathrm{f}, \mathrm{x}^{n+1:\infty}) \\
&= \iint \mathbb{1}\left\{g_a(\mathrm{x}, \mathrm{x}^{n+1:\infty}) \geq g_a(\mathrm{x}^{\text{test}}, \mathrm{x}^{n+1:\infty})\right\} dP_a(\mathrm{x} \mid \mathrm{f}, \mathrm{x}^{n+1:\infty}) dP_a(\mathrm{f}, \mathrm{x}^{n+1:\infty}) \quad \text{Lemma 3} \\
&= \iint \mathbb{1}\left\{g_a(\mathrm{x}, \mathrm{x}^{n+1:\infty}) \geq g_a(\mathrm{x}^{\text{test}}, \mathrm{x}^{n+1:\infty})\right\} dP_a(\mathrm{x} \mid \mathrm{x}^{n+1:\infty}) dP_a(\mathrm{x}^{n+1:\infty}) \\
&= \iint \mathbb{1}\left\{g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^{\infty}) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{x}^{\infty})\right\} d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x} \mid \mathrm{x}^{\infty}) d\mathrm{P}_{\boldsymbol{\theta}}(\mathrm{x}^{n+1:\infty} \mid \mathrm{x}^n) \\
&= p_{\text{mpc}}
\end{aligned}
$$

□

B Posterior predictive p-value algorithm

Algorithm 2 details the procedure for calculating the posterior predictive p-value in Equation (1) given train data x^n, test data x^{test}, a discrepancy function $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$, and the nuber of replication datasets to generate M.

Algorithm 2 $\widehat{p}_{\text{ppc}}$

Require: data $\{\mathrm{x}^n, \mathrm{x}^{\text{test}}\}$, discrepancy function $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{f})$, # replicates M

for $i \leftarrow 1$ to M **do**
  $\mathrm{f}_i \sim p_{\boldsymbol{\theta}}(\mathrm{f} \mid \mathrm{x}^n)$ ▷ sample explanation f
  $\mathrm{x}_i \leftarrow ()$ ▷ initialize replicant data
  for $j \leftarrow 1$ to n **do**
    $\mathrm{z}_j, \mathrm{y}_j \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} | \mathrm{f}_i)$ ▷ sample example from model likelihood
    $\mathrm{x}_i \leftarrow (\mathrm{x}_i, \mathrm{z}_j, \mathrm{y}_j)$ ▷ update replicant data
return $\frac{1}{\mathrm{M}} \sum_{i=1}^{\mathrm{M}} \mathbb{1}\left\{g_{\boldsymbol{\theta}}(\mathrm{x}_i, \mathrm{f}_i) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{f}_i)\right\}$ ▷ estimate p-value

C Lite generative predictive p-value algorithm

Algorithm 3 $\widehat{p}_{\text{gpc}}^{\text{lite}}$

Require: data $\{\mathrm{x}^n, \mathrm{x}^{\text{test}}\}$, a discrepancy function $g_{\boldsymbol{\theta}}(\mathrm{x}, \mathrm{x}^n)$, # replicates M

for $i \leftarrow 1$ to M **do**
  $\mathrm{x}_i \leftarrow ()$ ▷ initialize replicant data
  for $j \leftarrow 1$ to n **do**
    $\mathrm{z}_j, \mathrm{y}_j \sim p_{\boldsymbol{\theta}}(\mathrm{z}, \mathrm{y} \mid \mathrm{x}_i, \mathrm{x}^n)$ ▷ sample example from model
    $\mathrm{x}_i \leftarrow (\mathrm{x}_i, \mathrm{z}_j, \mathrm{y}_j)$ ▷ update replicant data
return $\frac{1}{\mathrm{M}} \sum_{i=1}^{\mathrm{M}} \mathbb{1}\left\{g_{\boldsymbol{\theta}}(\mathrm{x}_i, \mathrm{x}^n) \geq g_{\boldsymbol{\theta}}(\mathrm{x}^{\text{test}}, \mathrm{x}^n)\right\}$ ▷ estimate p-value

Algorithm 3 summarizes a "lite" version of the estimator that forgoes approximate sampling from the model posterior and likelihood. Instead, it samples replication data directly from the model predictive

distribution and calculates the discrepancy functions with respect to the observed data x^n rather than a dataset completion x^N.

D Additional figures

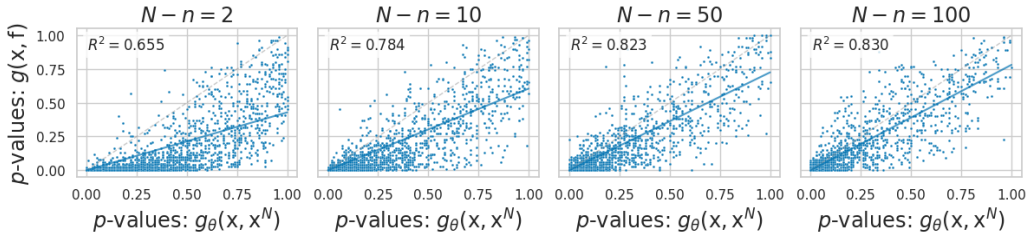
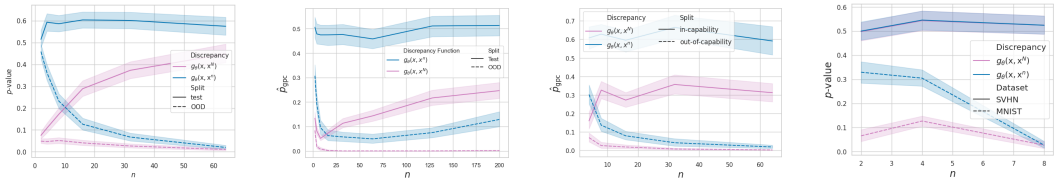


Figure 11: Scatter plots demonstrating that p_{gpc} becomes a better approximation of p_{ppc} with increasing dataset completion size $N - n$.

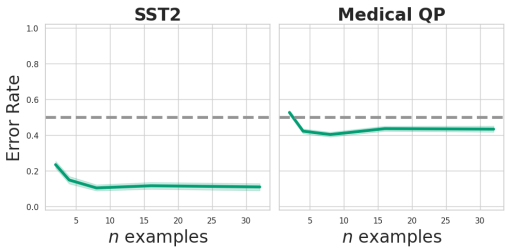


(a) Polynomial Tabular (b) ReLU-NN Tabluar (c) Natural language (d) Generative fill near

Figure 12: The generative predictive p-value against dataset size n

E Gemma-2 9B results

Figure 13a shows that the in-capability vs. out-of-capability distinction is also sensible for the Gemma-2 9B model. So we conduct the same analysis for Gemma-2 9B that we did for Llama-2 7B in Section 7.

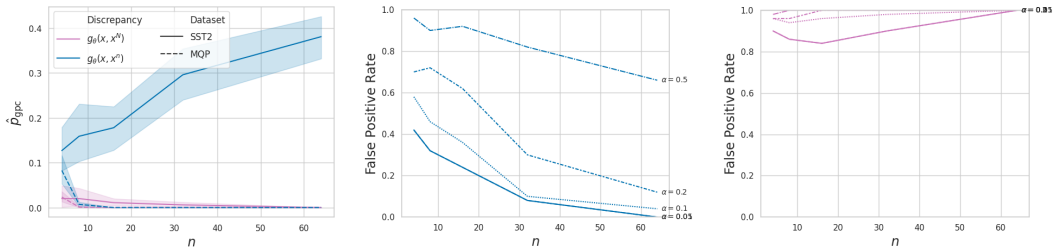


(a) Gemma 2 9B

Figure 13: Natural language in-capability vs. out-of-capability tasks.

Figure 14a shows p-values under the Gemma-2 9B model as a function of the ICL dataset x^n size n (context length). We see a clear separation between the estimated generative predictive p-values $\widehat{p}_{\text{gpc}}$ for the in-capacity SST2 data (solid lines) and the out-of-capacity MQP dataset (dashed lines), but only for the NLML discrepency. The separation is robust across different ICL dataset sizes.

Figure 14b plots the FPR for the capability detector defined by $\widehat{p}_{\text{gpc}}$ the with NLML discrepancy. We do not see the same stability of the FPR across ICL dataset size n that we saw for the Llama-2 7B model. Instead the FPR decreases with increasing n for all significance level α. Figure 14c plots the FPR for the capability detector defined by $\widehat{p}_{\text{gpc}}$ the with NLL discrepancy. We see that the false positive rate is high for all values. These findings are reflected in the Precision curves on the left



(a) Generative predictive p-values (b) False Positive Rate (NLML) (c) False Positive Rate (NLL)

Figure 14: Natural language task with Gemma-2 9B. The generative predictive p-value against dataset size n and it's relationship to the false positive rate. The first figure shows the generative predictive p-values, and the second and third figures show the false positive rate with the NLML and NLL discrepancy functions, respectively.

hand side of Figure 15. We again see in the Recall curve that the NLL discrepancy leads to a more sensitive predictor than the NLML discrepancy. The F1 and Accuracy curves show that the NLML based p-value leads to a much more effective predictor for Gemma-2 9B.

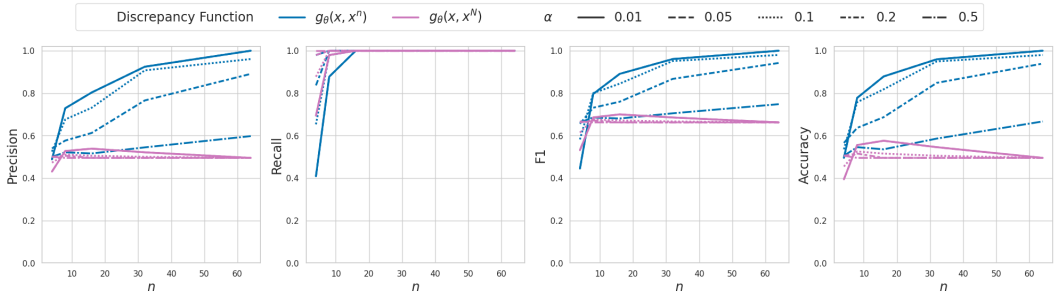


Figure 15: Natural language model suitability detection ablation. Precision, recall, F1, and accuracy metrics vs. number of in-context examples. SST2 ICL datasets are taken to be in-capability for Gemma-2 9b. MQP ICL datasets are taken to be out-of-capability for Gemma-2 9b.