On Evaluating LLMs' Capabilities as Functional Approximators: A Bayesian Perspective

Shoaib Ahmed Siddiqui*[1], Yanzhi Chen*[1], Juyeon Heo[1], Menglin Xia[2], and Adrian Weller[1,3]

[1]University of Cambridge
[2]Microsoft
[3]The Alan Turing Institute

Abstract

Recent works have successfully applied Large Language Models (LLMs) to function modeling tasks. However, the reasons behind this success remain unclear. In this work, we propose a new evaluation framework to comprehensively assess LLMs' function modeling abilities. By adopting a Bayesian perspective of function modeling, we discover that LLMs are relatively weak in understanding patterns in raw data, but excel at utilizing prior knowledge about the domain to develop a strong understanding of the underlying function. Our findings offer new insights about the strengths and limitations of LLMs in the context of function modeling.

1 Introduction

Large Language Models (LLMs) have revolutionized domains that can be formulated in a simple text-in-text-out format (Raffel et al., 2020; Brown et al., 2020). This includes a wide array of tasks from a helpful chatbot (Achiam et al., 2023), code assistant (Roziere et al., 2023), math theorem provers (Trinh et al., 2024), to an automated scientist (Lu et al., 2024).

Given the well-established performance characteristics of LLMs in a wide range of reasoning tasks (Achiam et al., 2023; Bubeck et al., 2023; Si et al., 2024; Lu et al., 2024), there has been growing interest in exploring their application on real-world prediction tasks i.e., as a regression system (Liang et al., 2022; Zheng et al., 2023; Roberts et al., 2023; Yu et al., 2023; Xiao et al., 2024). These approaches typically convert numerical data into tokens that can be processed by an LLM, which then predicts numerical targets for a query point conditioned on the task description and the provided in-context examples.

Prominent examples of using LLMs for function approximation include predicting velocities in robotics (Liang et al., 2022), predicting the accuracy of models with different configurations for architecture search (Zheng et al., 2023), directly predicting elevation given geospatial coordinates (Roberts et al., 2023), or even direct forecasting of time-series data (Yu et al., 2023; Requeima et al., 2024). Compared to conventional machine learning approaches, one immediate advantage of this LLM-based prediction paradigm is the natural ability to condition the model output on arbitrary side information (including detailed task description) provided in the form of natural language. This side information enables the language model to capture the underlying function accurately based on the rich prior knowledge it acquired in pretraining.

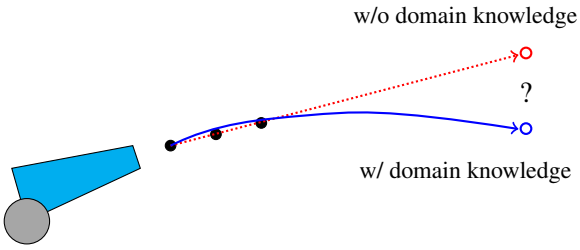


Figure 1: **A motivating example**. When making predictions, a model focusing only on raw data may interpret the underlying function as a linear one. However, when domain information is specified (i.e., the trajectory of a cannonball), the model can take into account physical laws for more accurate modeling of the trajectory. Given the vast amount of knowledge gathered during pretraining, LLMs can integrate domain knowledge they possess to generate more accurate predictions. We are interested in separately evaluating LLMs' ability of understanding raw data patterns and the ability of utilizing domain knowledge in function modelling tasks.

Despite the reported successes of LLMs in function modeling tasks, the underlying reasons for this performance remain poorly understood. Our work focuses on two key questions: (i) *can LLMs really comprehend patterns in raw data?*, and (ii) *to what extent can LLMs integrate and utilize domain-specific knowledge in function modeling?* As il-

*Authors contributed equally

Proceedings of the 31st International Conference on Computational Linguistics, pages 5826–5835
January 19–24, 2025.

lustrated in Fig. 1, domain knowledge can shape a strong prior for the underlying function and significantly affect prediction. Addressing these questions is crucial for the reliable and effective use of LLMs in real-world prediction tasks.

In order to help shed light on these questions, we present a novel evaluation framework for comprehensively assessing LLMs' function modeling capabilities. Our framework is inspired by principles from Bayesian machine learning, where we separate LLMs' function modeling abilities into two components: their ability to recognize patterns in raw data $\mathcal{D}$ (corresponding to the quality of the likelihood $p(\mathcal{D}|f)$, where f is the function to model) and their ability to incorporate domain knowledge possesses by the LLM (corresponding to the quality of the posterior $p(f|\mathcal{D}) \propto p(\mathcal{D}|f)p(f)$, with $p(f)$ the prior shaped by LLMs' domain knowledge). With this framework, we are able to pinpoint the strengths and weaknesses of state-of-the-art LLMs in function modeling tasks. Our contributions are:

- We propose a novel evaluation framework to comprehensively assess LLMs' function modeling capabilities, disentangling their ability to understand data patterns from their ability to incorporate prior domain knowledge.
- We evaluate the performance of state-of-the-art LLMs (e.g., GPT-4) on both synthetic and real-world prediction tasks, highlighting their strength and weaknesses as function approximators.

2 Background and Related Work

2.1 Large language models

Large Language Models (LLMs) are probabilistic models trained to predict the probability distribution over the next token conditioned on previous tokens (Radford et al., 2018) i.e.,

$$\text{LLM}_\theta(w_1, ..., w_t; \theta) := p(w_{t+1}|w_1, ..., w_t) \quad (1)$$

Outputs from the model are generated via autoregressive sampling of the next token $w_{t+1} \sim p_\theta(w_{t+1}|w_1, ..., w_t)$ conditioned on all the previous tokens (Radford et al., 2018). After training on massive datasets spanning trillions of tokens, the model acquires a vast amount of knowledge and reasoning capabilities (Bubeck et al., 2023; Dubey et al., 2024). This knowledge can be leveraged by the model for better function modeling e.g., by taking physical constraints into account when modeling a physical phenomenon (see Fig. 1).

2.2 LLMs as functional predictors

Given a prompt t describing the prediction task, systems that use LLMs to make prediction can be mathematically described as follows:

$$\hat{y}(x; t, \mathcal{D}) := \texttt{EXTRACT}(\arg\max_s p(s|x, t, \mathcal{D})) \quad (2)$$

where p is modeled via the language model LLM_θ, x is the query datapoint, t is the task description, $\mathcal{D}$ is the data used for in-context learning, and $\texttt{EXTRACT}(\cdot)$ extracts the prediction $\hat{y}$ from the generated sequence s [1]. Note that the in-context learning data can also be empty i.e., $\mathcal{D} = \varnothing$.

Prior work applying LLMs in different function modeling tasks can be seen as different instantiations of the same prediction framework. Liang et al. (2022) used LLMs to convert context-dependent terms such as 'more' or 'less' to exact velocity values. This can be seen as modeling the function between the terms and the velocity. Zheng et al. (2023) used GPT-4 to predict the accuracy of models with different configurations as a more efficient architecture search scheme, which can be seen as modeling the function behind hyperparameter configurations and accuracy. GPT4Geo (Roberts et al., 2023) attempted to leverage GPT-4 to directly predict elevation given geospatial coordinates, thereby also modeling a real-world function. Yu et al. (2023) evaluated the capabilities of LLMs to directly forecast financial time series. Similarly, Gruver et al. (2023) used LLMs to directly forecast time-series values while demonstrating their capability to model distributions. Requeima et al. (2024) introduced the idea of LLM Processes and applied LLM to a number of different time-series prediction tasks by conditioning on additional side information. Qin et al. (2023) conducted a comprehensive empirical assessment of GPT-4's capabilities across a spectrum of arithmetic reasoning tasks. Bubeck et al. (2023) also presented an example of function modeling tasks by evaluating the capabilities of GPT-4 in solving math riddles.

Unlike prior work that applied language models to novel function modeling tasks, we instead attempt to understand the reasons for their success by using a Bayesian evaluation framework, where we disentangle the language model's function modeling capabilities into two fundamental aspects.

[1] A regular expression parser can be sufficient in most cases, which requires the outputs of LLM to satisfy a certain format. This can be enforced by specifying it in the task description t.

Similar in spirit to our work, McCoy et al. (2024) also adapted a Bayesian perspective in studying LLMs' reasoning capabilities. However, unlike their work which used Bayesian principle to evaluate the impact of the occurrence of training data in purely numeric reasoning tasks, we use it to study the impact of LLM's prior domain knowledge when applied to real-world function modeling tasks. Furthermore, McCoy et al. (2024) did not consider the learning of functions from provided data — a task central to our work.

3 A Bayesian Evaluation Framework

Function modeling as Bayesian inference. We begin by framing the task of real-world function modeling as performing Bayesian inference in functional space (Ghahramani, 2015):

$$p(f|\mathcal{D}) \propto p(\mathcal{D}|f)p(f) \tag{3}$$

where $\mathcal{D}$ is the data, $p(\mathcal{D}|f)$ is the likelihood function that measures to what extent a function f matches with the data, and $p(f)$ is the prior over f. Both $p(\mathcal{D}|f)$ and $p(f)$ are important for accurate modeling of the function. For example, a linear function f_{linear} that accurately describes the trajectory of a ball in Fig. 1 may attain a high value of likelihood $p(\mathcal{D}|f)$, yet a good prior $p(f)$ based on physics would identify that a quadratic function $f_{\text{quadratic}}$ is indeed more plausible, given context information about the task.

Motivated by this Bayesian view of function modelling, we propose to factorize LLMs' function modeling capabilities into two key aspects:

- The ability to understand the patterns present in *raw* data. This corresponds to the quality of the likelihood function $p(\mathcal{D}|f)$ in Eq. 3.
- The ability to incorporate domain knowledge in order to better estimate the underlying function. This corresponds to the quality of the posterior $p(f|\mathcal{D})$ over the function in Eq. 3, with $p(f)$ being the prior shaped by the domain knowledge acquired by the LLM during pretraining.

The prior $p(f)$ itself can be viewed as a posterior over f after seeing the massive data $\mathcal{D}_{\text{web}}$ on internet during LLM pretraining: $p(f) = p(f|\mathcal{D}_{\text{web}})$.

Evaluation objectives. We are interested in separately evaluating the aforementioned two capabilities of language models in function modeling tasks. This separate evaluation, supported by our techniques detailed below, enable us to understand different aspects of the language model's capabilities. Here, we assess them by the prediction accuracy $\texttt{ACC}$ of the language model on the test set $\mathcal{D}_{\text{test}}$:

$$\texttt{ACC}(\mathcal{D}, t) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{test}}}\left[\mathbf{1}[\hat{y}(x; \mathcal{D}, t) = y]\right] \tag{4}$$

By carefully specifying the data $\mathcal{D}$ and the prompt t, we can either utilize or disregard domain knowledge provided by the language model during prediction, leading to the isolated evaluation of the aforementioned two capabilities of the model.

3.1 Evaluating the ability to understand raw data patterns

In this section, we develop techniques for evaluating the quality of the ability of LLM to understand the patterns in raw data. The key to this evaluation is to remove the influence of the prior $p(f)$, where we ensure that no domain knowledge can be utilized by the LLM in its prediction. We realize this through two important operations applied to the prompts: $\texttt{NUMERIZE}(\cdot)$ and $\texttt{DECONTEXTUALIZE}(\cdot)$.

Numerizing data. We first remove any information about domain by turning each data x in the original data $\mathcal{D}$, which is potentially in verbal form, into purely numerical values:

$$x \leftarrow \texttt{NUMERIZE}(\mathtt{x}), \quad \forall \mathtt{x} \in \mathcal{D}$$

where $\texttt{NUMERIZE} : \mathcal{S} \to \mathbb{R}^{\mathtt{d}}$ is an operation that to map a sentence $s \in \mathcal{S}$ from the sentence space $\mathcal{S}$ to a real-valued vector. For example, the sentence '{education=PhD, age=33}' describing the features of an individual will be transformed to a datum $z = \{1.0, 0.33\}$. Here, the values z_i are normalized, so that $z_i \in [0, 1]$. Normalization is introduced to avoid a LLM from inferring the semantic meaning of features according to their scale, range, and distribution[2]. Note that a similar operation is usually applied before feeding data into classical machine learning approaches due to their inherent inability to condition on arbitrary text.

Decontextualizing task description. Another operation is to remove any information about the domain or the context from the task description t:

$$t \leftarrow \texttt{DECONTEXTUALIZE}(t)$$

[2]As an example, consider z_1 to encode the age of individuals. After observing a large number of values of z_1 in $\mathcal{D}$, a powerful LLM may infer that this feature corresponds to the age from the distribution and the range of the feature values.

Prompts for evaluating $p(\mathcal{D}|f)$ (case w/o domain knowledge)

[Task description t]: `Below you are asked to perform a classification task. In this task, you will be given some numerical features, and your task is to predict the class (0 or 1) given the features. An intuitive guess rather than an accurate estimate is enough.`

`Below are a few examples that you can learn from. Your prediction is based on them.`

[Data $\mathcal{D}$]:

```
example 1, features: [1.0, 0.33], class 0
example 2, features: [0.4, 0.24], class 1
...
```

`Now predict the class for the data below:`

`example x, features: [0.6, 0.55], class=?`

Prompts for evaluating $p(f|\mathcal{D})$ (case w/ domain knowledge)

[Task description t]: `Below you are asked to predict whether the individual has a high income given some features that describe the individual's info. An intuitive guess rather than an accurate estimate is enough.`

`In this process, please actively utilize any domain knowledge you know about the task` .

`Below are a few examples that you can learn from. Your prediction is based on them.`

[Data $\mathcal{D}$]:

```
case 1, education=PhD, age=33, high income
case 2, education=B.A, age=24, low income
...
```

`Now predict the income for the case below:`

`case x, education=MSc, age=55, income=?`

Figure 2: Example prompt configurations for evaluating the quality of the likelihood $p(\mathcal{D}|f)$ and the posterior $p(f|\mathcal{D})$ encoded by the LLM in a function modeling task. When evaluating the posterior, a prompt (highlighted in color gray) is used to explicitly encourage the LLM to make use of domain knowledge regarding the task.

where $\texttt{DECONTEXTUALIZE} : \mathcal{S} \to \mathcal{S}$ is an operation to rewrite the task description. For example, consider the original task description in the ball trajectory prediction example where t can be something like 'we would like to predict the trajectory of a ball given its past trajectory'. The DECONTEXTUALIZE function rewrites this task description as 'this is a regression task where we predict y from x given some training data', stripping away any possible cues to leak domain information.

3.2 Evaluating the ability to incorporate domain knowledge

In this section, we focus on techniques for evaluating the ability of LLM to incorporate domain knowledge in function modeling tasks. Unlike the previous case where we remove the impact of prior, here we aim to emphasize the influence of the prior $p(f)$, which represents the domain knowledge the LLM holds. We achieve this by two operations applied to the prompts: $\texttt{VERBALIZE}(\cdot)$ and $\texttt{CONTEXUALIZE}(\cdot)$.

Verbalizing data. Our first operation corresponds to rewriting each data x in the original dataset $\mathcal{D}$ by 'verbalizing' it:

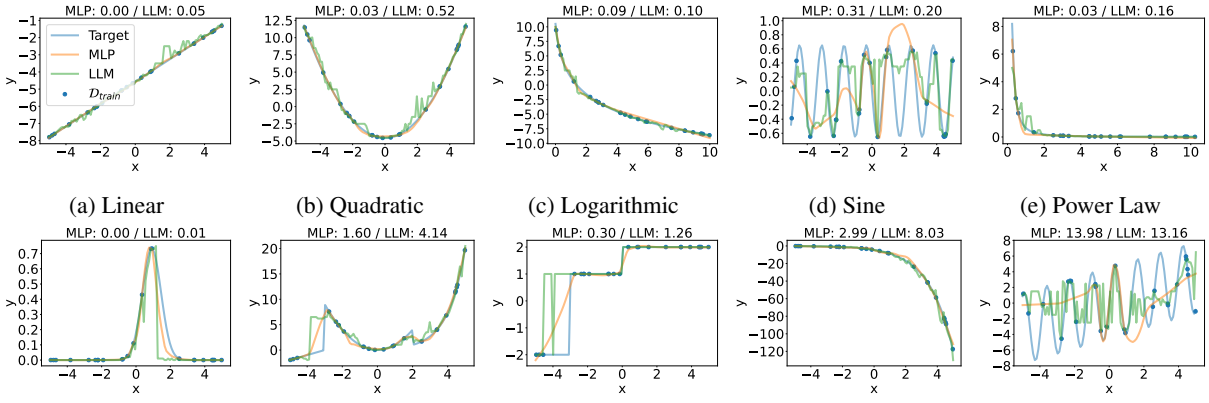
$$x \leftarrow \texttt{VERBALIZE}(\texttt{x}), \quad \forall \texttt{x} \in \mathcal{D}$$

where $\texttt{VERBALIZE} : \mathcal{S} \to \mathcal{S}$: is a function transforms all numerical features in the original data to its natural language-based representation. During this process, the semantic meaning of each feature will also be clearly indicated if they are not specified in the original data. For example, the sentence s='$\{1, $'married'$, 27\}$' will be rewritten as s'='{Gender=Female, Marriage status='married', Age=27}' to better convey the context of the data. This operation can be seen as the reverse operation of the previous $\texttt{NUMERIZE}(\cdot)$ operation.

Amplifying the impact of prior. Our second operation is to add context information to the task description as well as explicitly prompting the LLM to actively make use of any prior knowledge. Specifically, we rewrite the task description t as:

$$t \leftarrow (\texttt{CONTEXUALIZE}(t), h)$$

where $\texttt{CONTEXUALIZE}(\cdot) : \mathcal{S} \to \mathcal{S}$ is a function that adds context information of the task (e.g. domain, explanation of features, data source, etc.) and $h \in \mathcal{S}$ is an additional 'hinting' prompt concatenated to the contexualized prompt. This hinting prompt h is to trigger LLM to explicitly rely on domain knowledge it possesses when making prediction. One instance of h is 'during the process, please actively make use of any domain knowledge or prior information you know about [keywords] and incorporate it with the patterns you see from the data.', with [keywords] being the name of the domain e.g., law, physics, finance, medicine, etc.



(a) Linear (b) Quadratic (c) Logarithmic (d) Sine (e) Power Law

(f) Gaussian (g) Piecewise (h) Step (i) Exponential (j) Periodic Linear

Figure 3: Basic evaluations of function modeling using 25 training points, where we compare LLM performance (in particular GPT-4) with a 4-layer MLP with 64 hidden units. The MSE indicates direct prediction performance.

4 Experiments

In this section, we evaluate the function modeling capabilities of LLMs in both synthetic and real-world tasks. We mainly focus on GPT-4 for our experiments, which was the most capable model during our evaluations, though our evaluation can fully be adapted to any other more recent models.

4.1 Synthetic data

Setup. We first focus on evaluating LLMs' ability to understand patterns in raw data, where no domain knowledge is available. Here, we consider 10 types of commonly seen 1D functions. For each of these functions, we generate a total of 25 samples $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{25}$. We ask the language model to make predictions for different query points x' conditioned on the training examples.

Main results. In Fig. 3, we compare the performance of an LLM with that of a simple MLP. We find that while LLMs can approximately capture function shape in most cases, they struggle to provide smooth and accurate predictions generated by MLPs. This difference is further evidenced by the reported MSE values. A closer examination reveals that while LLMs can model simple functions such as linear and quadratic functions accurately, they struggle in modeling more complex functions such as periodic functions (Fig. 3d) or composite functions (Fig. 3j). This highlights that language models struggle to model functions directly from raw data (except in the simplest of cases), which might be under-represented in the pretraining dataset (McCoy et al., 2024). Such inability may also be attributed to the tokenization process in LLMs (Spathis and Kawsar, 2024).

4.2 Income prediction

Setup. We next consider a prediction task in social-economical study. The task here is to ask LLM to predict the income of an individual in the US given their personal information. The data x in this task consists of 13 features $x = \{x_1, ..., x_{13}\}$ describing e.g. the age, occupation and education of the individual, and is taken from the UCI Adult dataset (Becker and Kohavi, 1996). The binary target $y \in \{0, 1\}$ to predict is the income level of the individual (low v.s. high). Here the underlying function $f : \mathbb{R}^{13} \to \{0, 1\}$ is multi-dimensional.

Since UCI Adult is a widely used dataset, a LLM may have seen this data during pre-training (Oren et al.). To reduce the impact of potential test set memorization, which may lead to inaccurate evaluation of the language model's true function modeling capabilities, we modify the original dataset. Specifically, we (a) change the names of features (e.g. 'education' $\to$ 'degree'); (b) change feature values and scales by adding noise to the age feature and simplifying marital status to be binary; (c) replace features with equivalent counterparts e.g. 'hours per week' $\to$ 'hours per day'; (d) merge features such as merging capital gain/capital loss into capital net gain. This results in a new dataset that is not directly seen by the language model.

A total number of $n = 100$ samples are used for in-context learning.

Main results. In Table 1, we compare the predictive performance of LLM between the case when there is only raw data (denoted as LLM w/o domain) and the case when both the data and the domain information are provided (denoted as LLM w/ domain). We also highlight the performance

Table 1: Income prediction: comparison of direction prediction performance of GPT-4 when w/ and w/o domain knowledge. The performance of a 4-layer MLP with 500 units trained with n samples is also shown as reference.

	LLM w/o domain	**LLM w/ domain**	**MLP** ($n = 10^2$)	**MLP** ($n = 10^4$)
test accuracy (%)	62.6 ± 0.61	79.4 ± 0.83	73.9 ± 0.45	82.0 ± 0.52

of an MLP trained on the same data for reference. Our findings can be summarized as:

- *Raw data only:* The language model struggles to make accurate predictions in this case, as evident by the significant gap in performance as compared to the MLP. This highlights the difficulty in modeling complex multi-dimensional relationships just based on raw data using a language model. These results are consistent with our findings on the synthetic dataset.
- *With additional domain knowledge:* The performance of the language model conditioned on the domain knowledge improves significantly, which is on par with an MLP trained on two orders of magnitude more data.

The gap between the two cases highlights LLMs' efficacy in incorporating prior knowledge to update their understanding of the underlying function, compensating for their limitations in processing raw patterns. The results also underscore LLM's potential in small data regimes, where domain priors provide valuable compensation for limited data.

Further analysis. In addition to comparing prediction accuracy, we further employ two evaluation methods: prediction interpretation and feature selection to gain further insight into the differences in the model's interpretation of the underlying function f with and without domain knowledge.

In *prediction interpretation*, we ask the language model to output the rules used in prediction[3]. These rules reflect the language model's understanding of the underlying functions. Table 2 summarizes the rules for cases with and without domain knowledge respectively. Comparing the two sets of rules, we discover that when domain knowledge is available, LLM tends to make use of robust features that align well with common sense, whereas when there is no domain knowledge, it relies more on spurious features that may only be predictive for the provided in-context examples. This difference again highlights that state-of-the-art LLMs such as GPT-4 can effectively utilize domain knowledge to improve their function modeling capabilities.

In *feature selection*, we ask LLM to find the top-k features $X' \subset X$ which together as a whole are most informative about the target Y:

$$\max_{X':X'\subset X,|X'|=k} I(X';Y) \tag{5}$$

where $I(\cdot;\cdot)$ denotes mutual information. To select good features, one needs to understand both (a) the individual contribution of each feature to the underlying function and (b) how features interact with each other (e.g., synergy effects, redundancy, etc.). As such, this serves as a useful proxy for evaluating LLMs' function modeling capabilities. Table 3 summarizes the results. Leveraging its domain knowledge, GPT-4 is able to select a subset of features that closely matches the output of state-of-the-art feature selection methods (Yamada et al., 2020), while being two orders of magnitude more data efficient. In contrast, when relying solely on raw data, the model selects a poor set of features. These results highlight the impact of domain knowledge on LLM's function modeling process.

4.3 CO_2 emission level modeling

Setup. We further consider a function modeling task in climate science. In this task, we would like to ask LLM to predict the CO_2 concentration level $y \in \mathbb{R}$ given the time $x \in (1975, 2000)$. The data is collected in the Mauna Loa Observatory (Carbon Dioxide Research Group, 2004). The underlying function $f : \mathbb{R} \to \mathbb{R}$ is one-dimensional.

In order to reduce the impact of potential test set memorization (Oren et al.), we similarly employ techniques to update the dataset, including (a) hiding the information about the exact observatory that this data was collected from; (b) add random Gaussian noise $\epsilon \sim \mathcal{N}(\epsilon; 1, 10^{-2})$ to the measurements; and (c) shift the data by 1 unit, creating an unseen dataset.

We use data up to year 1980 as our training set (used via in-context learning), and use the data between year 1990 and year 1992 as our test set.

[3]This is similar in spirit to interpretability derived from chain-of-thought traces (Wei et al., 2022). Note that these explanations could be misaligned with the actual rules used by the model (Madsen et al., 2024) due to well-known problems regarding language model hallucination (Huang et al., 2023).

Table 2: Income prediction: Comparing the main prediction rules GPT-4 found when w and w/o domain knowledge.

	Rules found w/o domain	Rules found w/ domain
1	If feature 0 (age) and feature 2 (representative weight) are greater than 1, the sample is likely to be within class 1 (high income).	People who have higher education (Masters, Doctorate, Bachelors) tend to have higher income.
2	Higher values of feature 2 (representative weight) often correspond to Class 1, while values around 0 seem more correlated with Class 0 (low income).	Individuals with 'Married-civ-spouse' marital status are more correlated to higher income than individuals who are 'Never-married' or 'Divorced'.
3	If feature 8 (gender) is 0 and feature 0 (age) is below 0.5, the sample is likely Class 1.	Occupation such as 'Exec-managerial' and 'Prof-specialty' are better paid.

Table 3: Income prediction: Comparing the features selected by GPT-4 and that by state-of-the-art feature selection method. Accuracy is measured by an MLP trained using 10^4 samples with the selected features on the test set.

	LLM w/o domain ($n = 10^2$)	**LLM w/ domain** ($n = 10^2$)	(Yamada et al., 2020) ($n = 10^4$)
top 5 features	{gender, representative weight, age, hours per day, ethnicity}	{degree, martial status, occupation, hours per day, capital net gain}	{degree, martial status, age, hours per day, capital net gain}
test accuracy (%)	76.22 $\pm$0.56	82.63 $\pm$0.40	82.95 $\pm$0.38

Main results. In Fig. 4a, we compare the predictive performance of LLM on the test set between cases when there is only raw data (denoted as LLM w/o domain) and the case when both the data and the domain information are provided (denoted as LLM w/ domain). For reference, we also show the performance of a Gaussian process (GP) trained with the same data in Fig. 4b with an expert-chosen kernel taken from (Williams and Rasmussen, 2006). Consistent with prior results, domain knowledge plays a critical role in transforming a language model into an accurate functional approximator. To summarize:

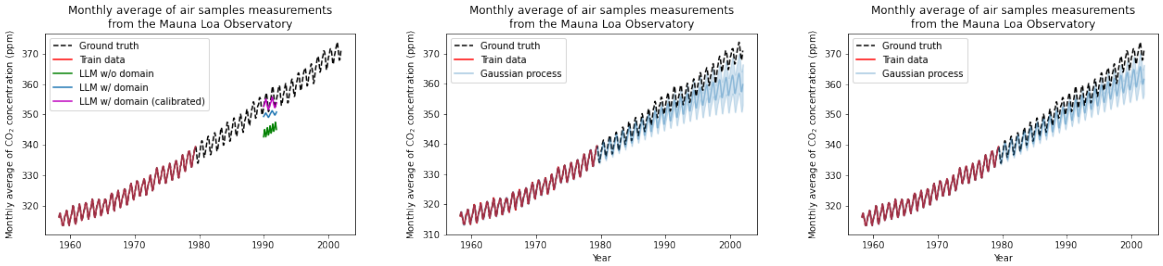
- *Raw data only:* When provided only with raw data, the model struggles to correctly model the concentration level. Specifically, it not only underestimates the CO_2 concentration level but also incorrectly models the frequency of the seasonal pattern, significantly underperforming compared to a GP trained on the same data. This result aligns with our earlier findings in Section 4.2, where we observed that the language model struggles to make accurate predictions for a similar 1D function (see Fig. 3j).
- *With additional domain knowledge:* We see a significant boost in model performance with the inclusion of domain knowledge. Both the CO_2 concentration level and the seasonal period exhibit significant improvements in comparison to just raw data. Notably, predictions made by the language model in this case are comparable to an expert-designed GP (shown in Fig. 4b).

The gap between the two cases again highlights the critical role of the prior in the function modeling capabilities of LLMs, resulting in predictions that are comparable to, or even surpass, those of an expert-designed Gaussian process model.

Further analysis. Similar to the evaluation in Section 4.3, we use prediction interpretation and kernel selection to gain better insights into the language model's function modeling process.

In *prediction interpretation*, we ask LLM to verbalize the rules it used for prediction[4]. Table 4 highlights the rules that the language model assumes to be using. For this task, both cases correctly recognize the two main types of patterns in the data i.e., the overall increasing trend and the seasonality. However, there is a significant improvement in modeling capability with domain knowledge. For instance, when modeling seasonality, domain knowledge helps pinpoint the exact peak and the duration of the cycle, making it more precise than the case with raw data alone. Crucially, with domain knowledge, the LLM can also detect patterns not directly apparent in the data, such as the increasing rate of change. When explicitly prompting the lan-

[4]See caveat presented in footnote 3.



(a) LLM prediction (b) GP (expert-designed kernel) (c) GP (LLM-designed kernel)

Figure 4: CO_2 level modeling: (a) Predictions made by GPT-4 with and without domain knowledge. (b-c) Predictions made by Gaussian processes with various kernels. The expert kernel is taken from (Williams and Rasmussen, 2006).

Table 4: CO_2 level modeling: Comparing the main prediction rules GPT-4 uses w/ and w/o domain knowledge.

	Rules found w/o domain	Rules found w/ domain
1	Increase Trend: Although Y has oscillations, there is a general upward trend. As X increases, the Y values tend to increase overall.	Growing Trend: There is a clear growth of CO_2 concentration from 1958 to 1975, aligning with established knowledge that human activities (e.g. fossil fuel burning, deforestation) and industry contribute to rising CO_2 levels.
2	Periodic Oscillations in Y: There seems to be a cyclical or periodic pattern in the Y values. For example, Y rises and falls several times, suggesting a wave-like behavior with peaks and troughs, even though X increases steadily.	Seasonality: There's a seasonal pattern. CO_2 concentrations peak during early Northern Hemisphere spring due to reduced plant growth, and they reach a minimum during early fall when plant growth peaks.
3	Amplitude of Oscillations Grows Over Time: The oscillations in Y seem to become more pronounced as X increases.	Increasing Rate of Change: Notably, the rate of CO_2 increase is accelerating. This corresponds with the 20th-century surge in industrial activity.

guage model to calibrate its prediction to account for this pattern, it produces estimates that outperform an expert-designed GP. These results clearly demonstrate that LLMs possess a strong prior of the underlying function, and can effectively leverage this prior to improve function modeling.

In *kernel design*, we further test the language model's understanding of the underlying function by asking the language model to design the Gaussian Process (GP) kernel, which reflects the assumptions about the underlying function being modeled[5] (Williams and Rasmussen, 2006). We evaluate whether LLMs like GPT-4 can come up with good kernels by incorporating domain knowledge. Given data and domain specification, the language model suggests the following kernel that achieves a comparable performance to an expert-designed kernel:

$$k_{llm}(t,t') = \sum_{l=1}^{4} k_l(t,t') \tag{6}$$

where k_1, k_2, k_3, k_4 are the RBF kernel, the exponential sine squared kernel, the rational quadratic kernel and the white noise kernel, respectively. These kernels correspond to (1) the long-term trend of CO_2 emission; (2) the yearly cyclical pattern; (3) the short-term fluctuations; and (4) observation noise and unmodeled factors respectively. The choice of these kernels by GPT-4 reflects a strong understanding of the underlying function when provided with both raw data and domain knowledge.

[5]A kernel measures the similarity between two inputs in functional space, thereby implicitly modeling the function.

5 Conclusion

In this work, we introduced a novel evaluation framework to systematically and quantitatively assess the function modeling capabilities of Large Language Models (LLMs). By disentangling their ability to understand raw data patterns from their ability to leverage prior knowledge, we identified both strengths and weaknesses of LLMs for function modeling tasks in comparison to conventional machine learning models including MLPs and GPs, namely (a) they struggle to understand functions based on just raw data, except for the simplest cases, and (b) their true strength lies in incorporating domain knowledge. Our research provides

a foundation for the reliable and effective applications of LLMs in real-world prediction tasks.

Our evaluation also highlights LLMs' potential as powerful functional predictors that outperform conventional ML models in some cases. For example, in cases with limited data, LLMs can effectively make use of domain knowledge to compensate for the scarcity of data (see Section 4.2). Furthermore, we find that LLMs can discover patterns not directly seen in the data by incorporating its domain knowledge (see Section 4.3). These abilities are not available in conventional machine learning models.

Our research suggests that future advancements in LLMs may benefit from explicitly enhancing their ability to understand raw data patterns during pretraining, which would significantly expand their applicability to numerical prediction tasks. Additionally, the development of powerful multimodal models that can simultaneously process numerical data and understand contextual information presents a promising research direction for future.

Limitation

While our evaluation techniques are broadly applicable to any language model, they are predominantly based on GPT-4 due to space constraints, as it was the most powerful model at the time of writing. Additionally, the experimental results may vary over time due to model updates. We therefore advise readers to interpret our findings cautiously, though our evaluation method can be fully reused for assessing future models.

Furthermore, our current evaluation focuses on the case of in-context learning, and is aligned with prior work in this space. We consider the evaluation of finetuned models to be an exciting avenue for the future.

Finally, as noted in the literature, LLM outputs are highly dependent on input prompts (Lester et al., 2021). Therefore, we assume the exact results can vary to a moderate extent based on the prompting technique used. However, we expect our findings to generalize across these different prompting techniques.

Acknowledgements

We thank the anonymous reviewers for their insightful feedback and suggestions. This project is supported by the *Microsoft Accelerate Foundation Models Research* (AFMR) program through which leading foundation models hosted by Microsoft Azure and access to Azure credits were provided to conduct the research. AW acknowledges support from a Turing AI Fellowship under grant EP/V025279/1, the Alan Turing Institute, and the Leverhulme Trust via CFI. YC acknowledges support from Cambridge Trust.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Barry Becker and Ronny Kohavi. 1996. Adult. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5XW20.

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

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Scripps Institution of Oceanography Carbon Dioxide Research Group. 2004. mauna-loa-atmospheric-co2. Weekly carbon-dioxide concentration averages derived from continuous air samples for the Mauna Loa Observatory, Hawaii, U.S.A., https://cdiac.ess-dive.lbl.gov/ftp/trends/co2/sio-keel-flask/maunaloa_c.dat.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Zoubin Ghahramani. 2015. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. 2023. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2022. Code as policies: Language model programs for embodied control. In *arXiv preprint arXiv:2209.07753*.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*.

Andreas Madsen, Sarath Chandar, and Siva Reddy. 2024. Are self-explanations from large language models faithful? In *Findings of the Association for Computational Linguistics ACL 2024*, pages 295–337.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Mathew D Hardy, and Thomas L Griffiths. 2024. Embers of autoregression show how large language models are shaped by the problem they are trained to solve. *Proceedings of the National Academy of Sciences*, 121(41):e2322420121.

Yonatan Oren, Nicole Meister, Niladri S Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. Proving test set contamination in black-box language models. In *The Twelfth International Conference on Learning Representations*.

Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is chatgpt a general-purpose natural language processing task solver? *arXiv preprint arXiv:2302.06476*.

Alec Radford, , Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

James Requeima, John Bronskill, Dami Choi, Richard E Turner, and David Duvenaud. 2024. Llm processes: Numerical predictive distributions conditioned on natural language. *arXiv preprint arXiv:2405.12856*.

Jonathan Roberts, Timo Lüddecke, Sowmen Das, Kai Han, and Samuel Albanie. 2023. Gpt4geo: How a language model sees the world's geography. *arXiv preprint arXiv:2306.00020*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can llms generate novel research ideas? a large-scale human study with 100+ nlp researchers. *arXiv preprint arXiv:2409.04109*.

Dimitris Spathis and Fahim Kawsar. 2024. The first step is the hardest: Pitfalls of representing and tokenizing temporal data for large language models. *Journal of the American Medical Informatics Association*, 31(9):2151–2158.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. 2024. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Christopher KI Williams and Carl Edward Rasmussen. 2006. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

Tim Z Xiao, Robert Bamler, Bernhard Schölkopf, and Weiyang Liu. 2024. Verbalized machine learning: Revisiting machine learning with language models. *arXiv preprint arXiv:2406.04344*.

Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. 2020. Feature selection using stochastic gates. In *International Conference on Machine Learning*, pages 10648–10659. PMLR.

Xinli Yu, Zheng Chen, Yuan Ling, Shujing Dong, Zongyi Liu, and Yanbin Lu. 2023. Temporal data meets llm–explainable financial time series forecasting. *arXiv preprint arXiv:2306.11025*.

Mingkai Zheng, Xiu Su, Shan You, Fei Wang, Chen Qian, Chang Xu, and Samuel Albanie. 2023. Can gpt-4 perform neural architecture search? *arXiv preprint arXiv:2304.10970*.