Do We Always Need Sampling? Eliciting Numerical Predictive Distributions of LLMs Without Auto-Regression

Anonymous Author(s)
Affiliation
Address
email

Abstract

Large Language Models (LLMs) have recently been successfully applied to regression tasks—such as time series forecasting and tabular prediction—by leveraging their in-context learning abilities. However, their autoregressive decoding process is ill-suited to continuous-valued outputs, and obtaining predictive distributions over numerical targets typically requires repeated sampling, leading to high computational cost. In this work, we investigate whether distributional properties of LLM predictions can be recovered without explicit autoregressive generation. To this end, we study a set of regression probes trained to predict statistical functionals (e.g., mean, median, quantiles) of the LLM's numerical output distribution directly from its internal representations. Our results suggest that LLM embeddings carry informative signals about numerical uncertainty, and that summary statistics of their predictive distributions can be approximated with reduced computational overhead. This investigation opens up new questions about how LLMs internally encode uncertainty in numerical tasks, and about the feasibility of lightweight alternatives to sampling-based approaches for uncertainty-aware numerical predictions.

1 Introduction

With the increasing capabilities of LLMs, a growing body of work has explored their use for structured data prediction–most notably for tabular data regression [e.g. Requeima et al., 2024, Hegselmann et al., 2023, Shysheya et al., 2025, Vacareanu et al., 2024] and time series forecasting [e.g. Gruver et al., 2024, Xue and Salim, 2023]. These studies demonstrate that LLMs can act as competitive regressors, even without task-specific fine-tuning. This advantage is especially pronounced in low-data regimes, where LLMs can leverage their pretraining, prior knowledge, and capacity to condition on auxiliary textual context to match or outperform specialised models.

However, issuing numerical predictions with LLMs remains computationally expensive due to their autoregressive nature: real-valued numbers typically span multiple tokens, and decoding them requires sequential auto-regressive generation. This is particularly problematic when apart from the point prediction one would like to also quantify the prediction uncertainty, which requires repeated sampling from the model's output distribution or auto-regressive computation of token logits [Gruver et al., 2024, Requeima et al., 2024].

This motivates us to explore whether the internal representations of pre-trained LLM's encode enough information to recover the entire predictive distribution–without resorting to autoregressive decoding. This is a non-trivial question; producing a complete number involves resolving its order of magnitude, which depends on decisions such as decimal placement or termination–often made only after several tokens have already been generated.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

Focusing on the problem of time series forecasting specifically, we explore to what extent the LLM's internal representation of the input sequence can be used to reconstruct its numerical predictive distribution of the next number. Concretely, we explore the following three questions:

Do LLMs encode the next number they intend to generate? (Section 3) We begin by examining whether LLM's internal representations encode sufficient information to recover point predictions–specifically, the greedy output, mean, and median of the predictive distribution. To test this, we develop a magnitude-factorised regression probe that separates prediction into two components: a coarse magnitude classification and a scale-invariant value regression, such that our model can effectively learn to predict numbers of varying orders of magnitude. Trained on LLM embeddings from synthetic time series data, *our probe accurately predicts numerical targets across data with varying orders of magnitude.*.

Can we elicit the uncertainty of the LLM's predictive distribution? (Section 4) We then ask whether uncertainty information is also captured in LLM's hidden states. Using quantile regression, we train probes to predict various quantiles of the LLM's output distribution, approximated via sampling. *The resulting models accurately recover the interquartile range, produce well-calibrated confidence intervals, and may allow to obtain sample-efficient estimates of statistical functionals.*

Do these results generalise to other settings? (Section 5) The ability to recover numerical predictions directly from LLM embeddings holds the potential to bypass auto-regressive sampling–offering substantial computational savings. However, for such probes to be practically useful, they must generalise beyond the specific conditions under which they were trained. We therefore evaluate whether a single probing model can be deployed across varied settings without retraining. First, we test generalisation to unseen time series lengths. Second, we assess generalisability of our previous results to real-world data. We investigate whether probes trained on real-world data generalise across different sub-domains and whether probes trained on synthetic data generalise to real-world data. *We demonstrate that, while some drop in calibration occurs on out-of-distribution datasets, our probes demonstrate encouraging generalisation abilities.*

Our findings provide new insights into the numerical capabilities of LLMs: much of the "reasoning" underlying numerical predictions appears to be encoded in the model's internal representations, prior to token-level decoding. This raises questions whether auto-regressive sampling is necessary to extract real-valued outputs from LLMs, and opens the door to developing more efficient single-pass approaches. By showing that both point estimates and uncertainty can be reliably extracted from hidden states, our work suggests a lightweight, general-purpose strategy for deploying LLMs in regression tasks–particularly in settings where computational efficiency and uncertainty estimation are essential. We hope these results motivate further study of how LLMs internally represent numerical quantities, and how this information can be surfaced for practical downstream use.

The code to reproduce our experiments can be found at https://anonymous.4open.science/r/guess_llm-811B/.

2 Related Works

Numerical Predictive Distributions of LLMs. When used as regressors, LLMs can provide not only point estimates but also full predictive distributions, reflecting their stochastic nature. To elicit continuous distributions over numerical outputs, Gruver et al. [2024] and Requeima et al. [2024] propose an autoregressive approach that generates logit values over discretised numeric bins, which are then scaled to form a valid probability distribution. Access to such distributions is crucial for downstream tasks requiring uncertainty quantification, including decision-making under uncertainty and Bayesian optimisation. However, these methods are computationally intensive, as they require multiple sequential queries to the LLM to construct a single distribution (e.g., $p(123.4) = p(1)p(2|1)p(3|12)p(.|123)p(4|123.)$). This motivates us to explore alternative approaches to eliciting numerical predictive distributions from LLMs.

Discrepancy between number generation and auto-regression. As next-token predictors, LLMs are not explicitly trained to understand the value of numbers. Due to their autoregressive nature, early tokens encode digits before key decisions like decimal placement (that determine a number's magnitude) are made. This can lead to surprisingly poor performance on simple numerical tasks [Yang et al., 2019, Akhtar et al., 2023, Zhou et al., 2024, Schwartz et al., 2024]. To address these limitations, several works have proposed alternatives to standard autoregressive decoding

for numerical predictions. For instance, Golkar et al. [2024] introduce a special [NUM] token, replaced post-hoc with a continuous value predicted by a learned regression head–though this requires retraining the model. Others [Singh and Strouse, 2024, Schwartz et al., 2024] investigate number-specific tokenizations to improve numerical accuracy of LLMs. In contrast, we ask whether one can bypass autoregressive decoding in *pre-trained* LLMs by directly reading out the predictive distribution from the internal representations.

Probing numeracy in LLM embeddings. A number of prior works give evidence that simple probing models can be used to learn numerical values encoded in the LLM embeddings. Wallace et al. [2019] has shown that the value of a number can be successfully decoded from its encoded word embedding (e.g., "71" $\rightarrow$ 71.0.). Stolfo et al. [2023] identified specific layers in LLMs that store numerical content, recoverable via simple linear probes, while Zhu et al. [2023] demonstrated that intervening on these layers alters generated outputs. More recently, Koloski et al. [2025] showed that LLM embeddings can serve as effective covariates in downstream regression models. Complementary findings from mechanistic interpretability suggest that, even in purely textual settings, LLM hidden states encode representations of tokens that the model is most likely to generate [Lindsey et al., 2025]. Taken together, these results support the hypothesis that it should be possible to train probes that approximate the numerical *predictive distribution* of the LLM, motivating our work.

3 Do LLMs Encode the Next Number They Intend to Generate?

LLMs are trained for next-token-predictions. Thus, as a single number typically spans multiple tokens, obtaining a complete numerical prediction from the LLM requires repeated auto-regressive sampling. This can be computationally expensive in number-heavy tasks, particularly when one would like to obtain repeated samples for the purpose of uncertainty estimation. To mitigate this overhead, we ask: *to what extent is the full predicted number–beyond just its leading digit–already encoded in the LLM's internal representation, prior to any token-by-token generation?* If such information can be reliably extracted, one could sidestep autoregressive generation altogether, enabling more efficient inference. However, this possibility is not trivial: critical aspects of number generation, such as the placement of the decimal point or number termination, which determine the order of magnitude of the number, often occur late in the decoding process, particularly for large magnitudes.

3.1 Method of Investigation

We provide an overview of our methodology below. For more details, see Appendix.

Objective. Let $\mathbf{x} = [x_1, \ldots, x_n]$ be a sequence of numbers (e.g., an equally-spaced time series). Given $\mathbf{x}$, a language model induces a predictive distribution $p_{\text{LLM}}(\cdot \mid \mathbf{x})$ over the next value x_{n+1}. In this section, we investigate whether the internal representations of the LLM encode sufficient information to predict this distribution's key statistics. Specifically, we aim to train independent probing models to recover: (a) the LLM's greedy prediction, (b) the mean of p_{LLM}, and (c) the median of p_{LLM}. We approximate the mean and median using 100 samples $y^j \sim p_{\text{LLM}}(\cdot \mid \mathbf{x})$.

LLM Representation. Let $\text{LLM}(\mathbf{x})$ denote the sequence of hidden states produced by the model when encoding $\mathbf{x}$, where, following Gruver et al. [2024], we serialise $\mathbf{x}$ to text as "$x_1, x_2, x_3, \ldots, x_n$,". We *do not* apply any scaling to the time series before serializing the inputs. This is important, as LLMs often contain contextual prior knowledge and scaling of the original time series may prohibit the LLM from using this prior knowledge effectively. From a pre-selected set of N layers $\mathcal{H}$, we extract the final token's hidden state from each layer, denoted $\mathbf{h}_\ell[-1] \in \mathbb{R}^{d_\ell}$. We concatenate these vectors to form a single input embedding for the probe:

$$\mathbf{e} := \text{concat}\left(\mathbf{h}_\ell[-1]\right)_{\ell \in \mathcal{H}} \in \mathbb{R}^{d_{\text{input}}}, \tag{1}$$

where $d_{\text{input}} = d_\ell \times |\mathcal{H}|$. The choice of the hidden layers in $\mathcal{H}$ is a hyperparameter of our model. Throughout this paper we use LLama-2-8B model for generating the embeddings, for which $d_\ell = 4096$. Experiments with other LLMs can be found in the Appendix.

Datasets. We use synthetically generated datasets to evaluate probing performance. Each sequence $\mathbf{x}$ is sampled from functions exhibiting varied numerical dynamics, including sinusoidal patterns, Gaussians, beat functions, and random noise (details in Appendix). The generated time series also vary in the length, n, and the level of noise, to ensure diversity in the generated embeddings. We generate variants of the dataset by rescaling the value range from $[-1, 1]$ to progressively larger intervals: $[-10, 10]$, $[-1000, 1000]$, and $[-10000, 10000]$. We then concatenate the datasets of different scales

to obtain a dataset of approximately 80k unique sequences, balanced across the different magnitudes. Our training datasets have the following structure: $\left\{\left(\mathbf{x}_i, \mathbf{e}_i, y_i^{\text{greedy}}, \{y_i^j\}_{j=1}^{100}\right)\right\}_{i=1}^{N}$, where N is the total number of examples in a dataset, y_i^j a single LLM sample from $p(\cdot|\mathbf{x}_i)$ and y_i^{greedy} the LLM's greedy prediction given $\mathbf{x}_i$.

Probing Model. The primary challenge in training regression probes for LLM numerical predictions lies in the wide spread of target magnitudes. Standard regression losses such as MSE or transformation techniques like log-scaling fail to provide stable gradients across this scale variability. To address this, we introduce a *magnitude-factorised regression model* that decomposes the prediction into a magnitude classification and a scale-invariant regression.

Let y^* be a target scalar (greedy, mean, or median prediction). We define its order of magnitude as:

$$m(y^*) := \lfloor \log_{10}(|y^*|) \rfloor . \tag{2}$$

Our model architecture consists of the following modules:

- $g : \mathbb{R}^{d_{\text{input}}} \to \mathbb{R}^{d_{\text{hidden}}}$: an encoder mapping the input $\mathbf{e}$ to a latent representation.
- $f_{\text{order}} : \mathbb{R}^{d_{\text{hidden}}} \to \mathbb{R}^M$: a classifier predicting logits over M magnitude bins.
- $f_{\text{val}} : \mathbb{R}^{d_{\text{hidden}}} \to \mathbb{R}$: a regressor predicting a rescaled value.

The final prediction is reconstructed by taking the expectation over the top-K predicted orders:

$$\hat{y}_i = \hat{r}_i \cdot 10^{\hat{m}_i}, \quad \text{where} \quad \hat{r}_i = f_{\text{val}}(g(\mathbf{e}_i)), \quad \hat{m}_i := 10^{\sum_{k \in \mathcal{K}_i} k \cdot p_k(g(\mathbf{e}_i))}, \tag{3}$$

where $\mathcal{K}_i$ is a set of K exponents with the largest logit values as predicted by $f_{\text{order}}(g(\mathbf{e}_i))$ and $p_k(g(\mathbf{e}_i))$ is derived from $f_{\text{order}}(g(\mathbf{e}_i))$ using the softmax over the top-K logits.

Training Objective. To decouple magnitude errors from value regression during early training, we use the ground-truth magnitude $m(y^*)$ to compute $\hat{y}$ and define the training objective as:

$$\mathcal{L} = \mathcal{L}_{\text{order}} + \beta \cdot \mathcal{L}_{\text{val}}, \quad \text{where} \tag{4}$$

$$\mathcal{L}_{\text{order}} = \frac{1}{N_b} \sum_{i=1}^{N_b} \text{CrossEntropyLoss}(\hat{m}_i, m(y_i^*)), \quad \mathcal{L}_{\text{val}} = \frac{1}{N_b} \sum_{i=1}^{N_b} \left(\hat{r}_i - \frac{y_i^*}{10^{m(y_i^*)}} \right)^2 . \tag{5}$$

In the above N_b is the batch size and the hyperparameter β balances the magnitude and value objectives. This formulation of the loss allows the model to learn scale-invariant value predictions and as we found out, it improves stability during optimization.

3.2 Results

Order of magnitude. We first investigate to what extent our probing model can correctly recover the order of magnitude of the number generated by the LLM. We train three separate models, for each of the mean, median and greedy predictions. As visualised on Figure 2, we find strong correlation between the $\log_{10}$ of the predicted number and $\log_{10}$ of the true value, for all three statistics. Further, the bar chart on the right hand side of Figure 2 visualises that our model achieves above 80% accuracy in predicting the exponent of the generated number.

Precision in generated digits. To further assess whether the LLM's internal representations encode fine-grained information beyond the order of magnitude, we focus on the dataset with time series values in the interval $[-1, 1]$. We report the mean squared error (MSE) of our predictions in Figure 1 and compare them against three baselines: $\bar{\mathbf{x}}$ (predicting the average value in the whole training dataset), $\bar{\mathbf{x}}_i$ (predicting the average of each time series) and $x_{i,n}$ (predicting the last value from each time series). We further plot the obtained predictions in Figure 3. Interestingly, among the three targets considered (mean, median, greedy), the model performs worst when predicting the greedy output. As shown in Figure 3, the probe captures the sign of the greedy prediction reliably but exhibits larger errors in the decimal digits. We hypothesise that this is because the greedy prediction is not an explicit function of the model's predictive distribution, but rather a byproduct of the autoregressive decoding process, making it harder to recover precisely from internal states.

Figure 1: MSE of the predicted values (no scaling).

target	$\hat{y}$ (ours)	$\bar{\mathbf{x}}$	$\bar{\mathbf{x}}_i$	$x_{i,n}$
mean	0.009	0.256	0.035	0.085
median	0.009	0.260	0.041	0.087
greedy	0.024	0.273	0.065	0.109

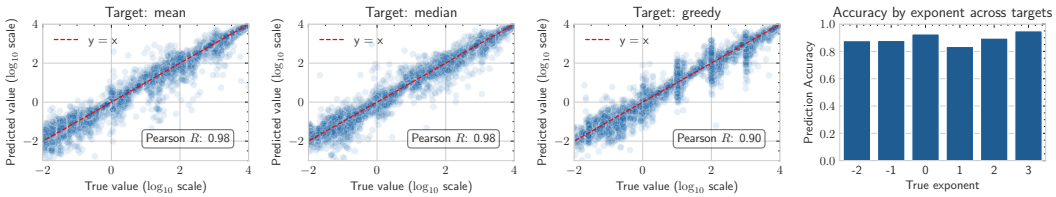


Figure 2: Predicted vs. true values of mean, median and greedy prediction, presented on $\log_{10}$ scale. The probing model accurately recovers the number that the LLM intends to predict, indicating that the internal representations encode the order of magnitude of prediction.

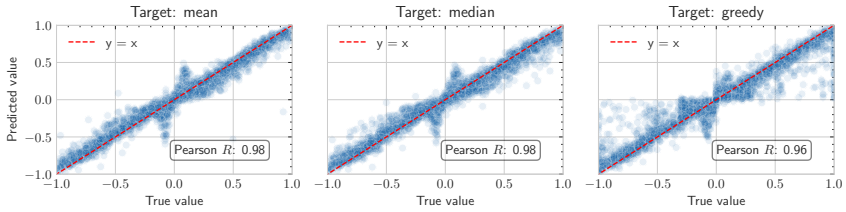


Figure 3: Predicted vs. true values of mean, median and greedy prediction. The probing model accurately recovers the number the LLM intends to predict, with the precision surpassing just order of magnitude estimation.

These results show that the internal representations of a pre-trained LLM encode detailed information about its intended numerical output–even before any tokens are generated. Our probing model accurately recovers not only the order of magnitude, but also fine-grained point estimates of the mean, median, and the greedy output. This demonstrates that much of the numerical reasoning performed by the LLM is already present in its hidden states, and may not require the autoregressive decoding process.

4 Can We Elicit the Uncertainty of the LLM's Predictive Distribution?

In the previous section, we demonstrated that point estimates—such as the greedy prediction, mean, and median—of an LLM's predictive distribution $p_{\text{LLM}}(\cdot \mid \mathbf{x})$ can be recovered from its internal representations without the need for performing auto-regressive sampling. Encouraged by these findings, we now investigate whether we can go beyond point estimates to recover the *uncertainty* of p_{LLM} by approximating its distributional shape. Specifically, we attempt to recover multiple quantiles of p_{LLM}, enabling a coarse-grained reconstruction of its distribution function an an easy way of estimating the confidence intervals for the LLM's predictions.

4.1 Method of Investigation

Why Quantiles? Since the underlying distribution p_{LLM} may be multi-modal and non-Gaussian, we rule out parametric approximations (e.g., fitting a Gaussian). Instead, we adopt *quantile regression*, which enables direct estimation of distributional shape without strong assumptions about its form.

Quantile Regression. Let $\mathcal{Q} = [\tau^1, \ldots, \tau^S]$ be a fixed list of target quantiles. For each quantile level $\tau^s \in [0, 1]$, we denote the predicted value as $\hat{q}^s$. We train the quantile predictor using the *pinball loss* [Koenker and Hallock, 2001], computed with respect to LLM samples $y_i^j \sim p_{\text{LLM}}(\cdot \mid \mathbf{x}_i)$. For a single quantile level τ, predicted quantile $\hat{q}$ and an LLM sample y, this loss function is defined as:

$$\text{PinballLoss}(\tau, \hat{q}, y) := \max\left(\tau(y - \hat{q}), (1 - \tau)(\hat{q} - y)\right). \tag{6}$$

Architecture. As in Section 3, we use a magnitude-factorised model to address the challenge of scale variance in numerical outputs. The model is defined as follows:

- $g : \mathbb{R}^{d_{\text{input}}} \to \mathbb{R}^{d_{\text{hidden}}}$: a shared encoder that maps the input representation $\mathbf{e}$ to a latent space.
- For each quantile index $s \in \{1, \ldots, S\}$:
  - $f^s_{\text{order}} : \mathbb{R}^{d_{\text{hidden}}} \to \mathbb{R}^M$: a classifier predicting the order of magnitude m^s of quantile q^s.

– $f^s_{\text{val}} : \mathbb{R}^{d_{\text{hidden}}} \to \mathbb{R}$: a regressor predicting a scale-invariant value $\hat{r}^s$.

Similarly as before, each quantile is reconstructed from the predicted components as:

$$\hat{q}^s_i = \hat{r}^s_i \cdot 10^{\hat{m}^s}, \quad \text{where} \quad \hat{r}^s_i = f^s_{\text{val}}(g(\mathbf{e}_i)), \quad \hat{m}^s_i = 10^{\sum_{k \in \mathcal{K}^s} k \cdot p^s_k(g(\mathbf{e}_i))}. \tag{7}$$

where $\mathcal{K}^s_i$ is a set of K exponents with the largest logit values as predicted by $f^s_{\text{order}}(g(\mathbf{e}_i))$ and $p^s_k(g(\mathbf{e}_i))$ is derived from $f^s_{\text{order}}(g(\mathbf{e}_i))$ using the softmax over the top-K logits.

Training Objective. As before, we use the true order of magnitude $m(y^j_i)$ for each target value during training to enable stable learning. The total loss is the sum of the cross-entropy losses for magnitude prediction and pinball losses for quantile regression:

$$\mathcal{L} = \sum_{s=1}^{S} w_s \left(\mathcal{L}^s_{\text{order}} + \beta \cdot \mathcal{L}^s_{\text{val}} \right), \tag{8}$$

$$\mathcal{L}^s_{\text{order}} = \frac{1}{N_b} \sum_{i=1}^{N_b} \text{CrossEntropyLoss}(f^s_{\text{order}}(g(\mathbf{e}_i)), m(y^*_i)), \tag{9}$$

$$\mathcal{L}^s_{\text{val}} = \frac{1}{N_b N_{sa}} \sum_{i=1}^{N_b} \text{PinballLoss}\left(\tau^s, \hat{r}^s_i, \frac{y^j_i}{10^{m(y^j_i)}} \right), \tag{10}$$

where N_b is the batch size, N_{sa} is the number of LLM samples per input, S is the number of quantiles, and $[w_1, \ldots, w_S]$ a set of weights per each quantile, which is a hyperparameter of our model. In our experiments we have $N_{sa} = 100$ and $S = 7$, with the corresponding quantile list $\mathcal{Q} = [0.025, 0.05, 0.25, 0.5, 0.75, 0.95, 0.975]$. This choice of quantiles allows us to estimate: the median, the interquartile range (IQR), as well as the 90% and 95% confidence intervals.

Datasets We use the same datasets as in section 3, considered separately rather than concatenated.

4.2 Results

IQR Prediction. To investigate whether the LLM's internal representations encode information about the spread of its predictive distribution, we estimate the interquartile range (IQR) using the predicted 25th and 75th percentiles. As the IQR is sensitive to scale, we normalise it by the predicted median, and similarly normalise the empirical IQR from LLM samples using the sample median. If the probe captures uncertainty faithfully, we should observe a monotonic relationship between the predicted and empirical (normalised) IQRs. As shown in Figure 4, we find a strong correlation between predicted and sample-based IQRs, suggesting that the model is able to infer distributional spread from internal LLM activations.

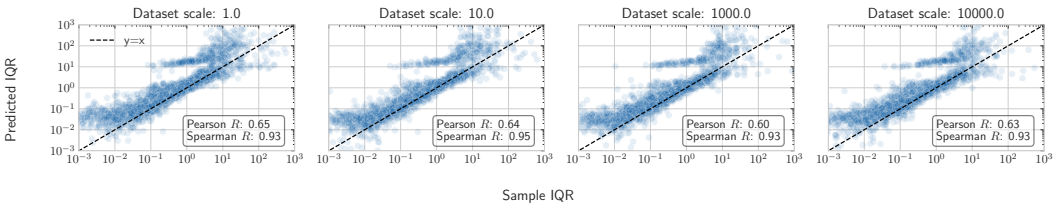


Figure 4: Predicted vs. sample-based IQR (both median-normalised). The model accurately tracks the variability of the LLM's output distribution.

Confidence Interval Coverage. We next evaluate whether the predicted quantiles yield calibrated confidence intervals. Given a desired confidence level α and its associated interval $\mathcal{C}(\alpha)$ predicted by the probe, we compute the empirical coverage by checking what fraction of LLM samples fall within the predicted interval. We expect that:

$$\alpha = \mathbb{E}_{y \sim p(\cdot|\mathbf{x})} \left[\mathbb{1}\{y \in \mathcal{C}(\alpha)\} \right] \approx \frac{1}{N_{sa}} \sum_{j=1}^{N_{sa}} \mathbb{1}\{y^j \in \mathcal{C}(\alpha)\}, \quad \text{where } y^j \sim p_{\text{LLM}}(\cdot|\mathbf{x}).$$

Table 1: Coverage of the predicted confidence intervals. Values denote empirical coverage (%) ± standard error.

α	50%	90%	95%
1.0	49.2 ± 0.4	89.2 ± 0.3	94.1 ± 0.3
10.0	49.8 ± 0.4	90.2 ± 0.3	94.1 ± 0.3
1000.0	50.4 ± 0.5	89.0 ± 0.3	93.7 ± 0.3
10000.0	51.2 ± 0.5	88.2 ± 0.4	92.7 ± 0.3

Table 1 reports the empirical coverage for 50%, 90%, and 95% intervals across datasets with different scaling parameters ℓ. In all cases, empirical coverage closely matches the target level, indicating that the quantile probe is well-calibrated.

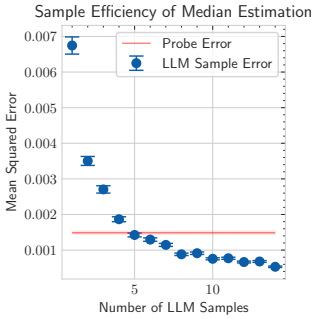


Figure 5: The probe (horizontal line) achieves lower MSE than sampling for $n \leqslant 5$.

Sample Efficiency. Finally, we examine whether the probe can outperform direct sampling from the LLM in terms of sample efficiency. Let S denote a target statistic (e.g., median or a quantile). We define $S(n)$ as the estimate from $n < 100$ samples and we let $S^* := S(100)$ as a proxy for the ground-truth. We compute the LLM sample error as $\text{MSE}(S(n), S^*)$ and compare it to the probe error $\text{MSE}(\hat{S}, S^*)$.

Figure 11 illustrates this comparison for the median on a dataset with scale 1.0. The probe outperforms empirical sampling for all $n \leqslant 5$, demonstrating that our approach can be more sample-efficient. Results for additional quantiles and larger-scale datasets are reported in the Appendix. A probe of this kind can serve as a computationally efficient surrogate for estimating statistics of the LLM output distribution which can help in cost and compute time reduction.

Key Insights.

- Our findings in this section provide strong evidence that the uncertainty of an LLM's predictive distribution is encoded in its internal activations and can be effectively elicited using a quantile regression probe. The probe is capable of predicting meaningful spread measures (e.g., IQR), producing well-calibrated confidence intervals that match the empirical coverage observed when generating samples from the LLM.
- Furthermore, the probe demonstrates an encouraging sample efficiency, enabling uncertainty estimation without incurring the cost of repeated sampling.
- Together, these results suggest that LLMs internalise rich distributional information during generation, which can be accessed and approximated efficiently via probing techniques. This opens up new opportunities for downstream applications that rely on uncertainty quantification—such as safe decision-making, model-based control, and probabilistic reasoning—while avoiding the overhead of sampling-based inference.

5 Generalisation Properties

In this section, we investigate the generalisation capabilities of our approach along several axes including context length generalisation, applicability to real-world data, and cross-dataset generalisation. As the process of training a probe can be costly, such generalisation capabilities are important for real-world applications, if we would like to use a pre-trained probe on new datasets with different distributional properties instead of performing repetitive auto-regressive sampling to estimate the LLM's predictive distribution.

Throughout this section, we use the same probing architecture introduced in Section 4.

5.1 Generalisation to Unseen Context Lengths

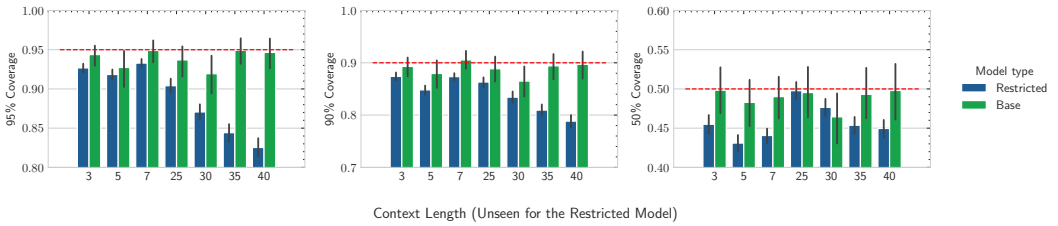


Figure 6: Generalisation to unseen context lengths. A probe trained on a restricted context length range (Restricted) exhibits greater deviation in empirical coverage outside its training range.

We begin by asking whether a probe trained on a fixed range of input sequence lengths generalises to longer or shorter contexts. We train and compare against each other two models:

- **Base:** Trained on input sequences $\mathbf{x}$ with length in the range $[3, 40]$.
- **Restricted:** Trained only on input sequences $\mathbf{x}$ with length in the range $[10, 20]$.

At test time, we evaluate both models on contexts shorter than 10 and longer than 20. We assess generalisation by measuring the empirical coverage of predicted confidence intervals, as defined in Section 4.2. Figure 6 shows results on the dataset with scale factor 1.0.

We observe that while both models achieve reasonable calibration, the Restricted model exhibits slightly greater deviations from the nominal coverage, particularly for context lengths further from the training distribution. These results suggest that the probe generalises to novel context lengths, but training on a wider context ranges should be beneficial for more robust generalisation.

5.2 Application to Real-World Data

Thus far, our analysis has focused on synthetic data. We now evaluate whether our probing model can be trained successfully on real-world datasets, and how well predictions can generalise across different types of input series.

To assess this, we construct a dataset using time series from the Darts [Herzen et al., 2022] and Monash [Godahewa et al., 2021] collections. Following the same format as in our synthetic experiments, we generate LLM embeddings and samples from their predictive distribution for approximately 45,000 distinct sequences across 32 sub-datasets (e.g., US Births, Bitcoin, Air Passengers). Furthermore, we also investigate an even stronger form of generalisation: from a model trained on synthetic data only to testing on real-world data. Thus, we train the following models:

- **Real (all):** Trained on a random 80% of all sequences across all sub-datasets. The remaining 20% is held out for testing.
- **Real (5 fold):** We partition the dataset into 5 folds such that, in each fold, one model is trained on 80% of the sub-datasets and evaluated on the remaining 20%. This ensures that each sub-dataset appears in the test fold of exactly one out of 5 models trained.
- **Synth:** A model trained on the combination of the 4 synthetic dataset from the previous sections with scales 1.0, 10.0, 1000.0 and 10000.0.

At test time, the above models face increasingly stronger distribution shifts. In terms of generalisation performance to previously unseen data distributions, we can view the Real (all) model as a baseline for Real (5 fold) and Synth.

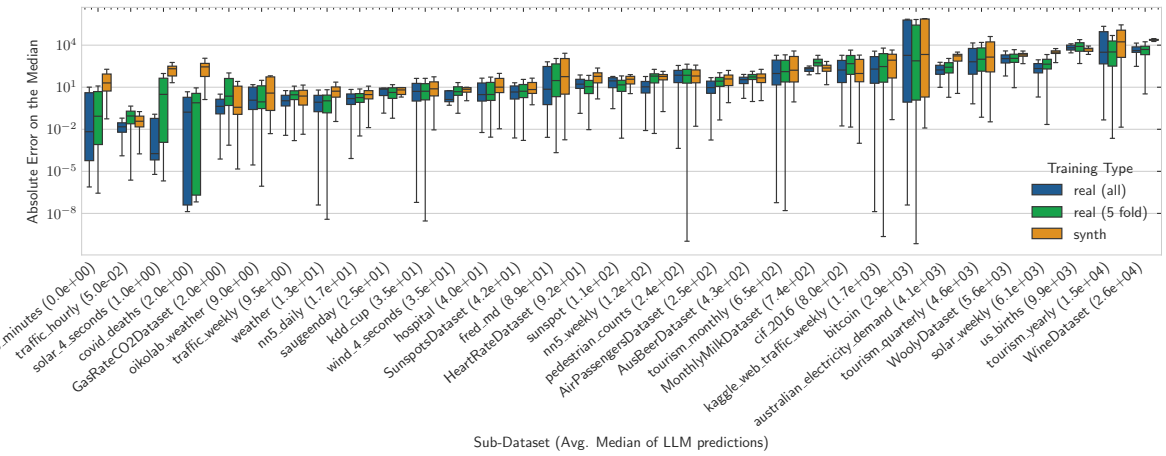
Table 2: Coverage of the CI intervals on previously unseen testing inputs.

Model \ α	50%	90%	95%
Real (all)	53.1 ± 0.5	89.4 ± 0.3	93.3 ± 0.3
Real (5 fold)	46.6 ± 0.2	83.7 ± 0.1	89.0 ± 0.1
Synth	40.7 ± 0.5	56.4 ± 0.6	67.4 ± 0.6

Firstly, in Table 2 we report the average coverage of the CI across all training types. We observe that the Real (all) model demonstrates good performance, with the empirical coverage of LLM samples closely matching the expected coverage. The Real (5 fold) model demonstrates a slight downgrade in performance. Interestingly, while the Synth model underperforms it still demonstrates good generalisation for some of the sub-datasets as we can see on Figure 7. This figure shows the distribution of the Absolute Error of the predicted median vs. the median of LLM samples across all sub-datasets. The x-axis is sorted by increasing order of magnitude of the datasets defined by the average of the median of LLM samples. We note that the sub-datasets in our collection cover widely varying ranges of values (with individual LLM samples varying in magnitude from 10^{-3} to 10^{13}). We suspect that this is the main reason for our probing model struggling to generalise across some datasets.

Key Insights.

- The probing model exhibit some, albeit limited, generalisation across context lengths.
- When applied to real-world datasets, the model achieves accurate empirical coverage and demonstrates partial transferability to unseen data distributions. Cross-dataset generalisation is possible, but challenged by large variation in scale and distribution.



task

Figure 7: Absolute Error on the Median across different sub-dataset. Comparison of generalisation across models trained on different data.

6 Discussion, Limitations and Further Work

Discussion Our findings demonstrate that LLMs internally encode rich numerical information about their intended predictions, well before any autoregressive decoding occurs. By training lightweight probes on hidden states, we can recover both point estimates (mean, median, greedy outputs) as well as informations about the uncertainty of the model's predictive distribution. This suggests that much of the LLM's "reasoning" over numeric outputs is already complete at the point of processing the input sequence, and that decoding primarily serves as a mechanism for surfacing the LLM's predictions. Beyond shedding light on the internal mechanics of LLMs in regression settings, these results open up a practical direction: enabling uncertainty-aware numerical prediction without incurring the high cost of repeated sampling.

Limitations Despite these promising findings, several limitations remain. First, while our approach applies to pre-trained model and does not require any fine-tuning, we assume access to internal model activations. Second, while our probing models exhibit some generalisation abilities, they are still trained per-model and require retraining for new architectures or tokenization schemes. Third, for training and evaluation purposes, we approximate the LLM's predictive distribution using empirical sampling, which is itself a noisy and computationally costly proxy.

Further Work Future research could explore extending this framework to broader classes of structured data and more diverse prediction tasks, including multivariate time series, univariate or multivariate regression tasks. A deeper investigation into the mechanistic basis of numerical encoding—i.e., how and where numerical quantities are represented across LLM layers—could also reveal connections to known computational circuits or arithmetic operations within the model. Finally, motivated by our generalisation results, an important next step is the development of a universal probing model which, for a given LLM, can be applied off-the-shelf across diverse tasks and data domains. This would eliminate the need for repeated retraining—an important consideration given the cost of training high-capacity probes at scale.

References

M. Abdin, J. Aneja, H. Awadalla, A. Awadallah, A. A. Awan, N. Bach, A. Bahree, A. Bakhtiari, J. Bao, H. Behl, A. Benhaim, M. Bilenko, J. Bjorck, S. Bubeck, M. Cai, Q. Cai, V. Chaudhary, D. Chen, D. Chen, W. Chen, Y.-C. Chen, Y.-L. Chen, H. Cheng, P. Chopra, X. Dai, M. Dixon, R. Eldan, V. Fragoso, J. Gao, M. Gao, M. Gao, A. Garg, A. D. Giorno, A. Goswami, S. Gunasekar, E. Haider, J. Hao, R. J. Hewett, W. Hu, J. Huynh, D. Iter, S. A. Jacobs, M. Javaheripi, X. Jin, N. Karampatziakis, P. Kauffmann, M. Khademi, D. Kim, Y. J. Kim, L. Kurilenko, J. R. Lee, Y. T. Lee, Y. Li, Y. Li, C. Liang, L. Liden, X. Lin, Z. Lin, C. Liu, L. Liu, M. Liu, W. Liu, X. Liu, C. Luo, P. Madan, A. Mahmoudzadeh, D. Majercak, M. Mazzola, C. C. T. Mendes, A. Mitra, H. Modi, A. Nguyen, B. Norick, B. Patra, D. Perez-Becker, T. Portet, R. Pryzant, H. Qin, M. Radmilac, L. Ren, G. d. Rosa, C. Rosset, S. Roy, O. Ruwase, O. Saarikivi, A. Saied, A. Salim, M. Santacroce, S. Shah, N. Shang, H. Sharma, Y. Shen, S. Shukla, X. Song, M. Tanaka, A. Tupini, P. Vaddamanu, C. Wang, G. Wang, L. Wang, S. Wang, X. Wang, Y. Wang, R. Ward, W. Wen, P. Witte, H. Wu, X. Wu, M. Wyatt, B. Xiao, C. Xu, J. Xu, W. Xu, J. Xue, S. Yadav, F. Yang, J. Yang, Y. Yang, Z. Yang, D. Yu, L. Yuan, C. Zhang, C. Zhang, J. Zhang, L. L. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, and X. Zhou. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone, Aug. 2024. URL http://arxiv.org/abs/2404.14219. arXiv:2404.14219 [cs].

M. Akhtar, A. Shankarampeta, V. Gupta, A. Patil, O. Cocarascu, and E. Simperl. Exploring the Numerical Reasoning Capabilities of Language Models: A Comprehensive Analysis on Tabular Data. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15391–15405, Singapore, Dec. 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.1028. URL https://aclanthology.org/2023.findings-emnlp.1028/.

R. Godahewa, C. Bergmeir, G. I. Webb, R. J. Hyndman, and P. Montero-Manso. Monash Time Series Forecasting Archive, May 2021. URL http://arxiv.org/abs/2105.06643. arXiv:2105.06643 [cs].

S. Golkar, M. Pettee, M. Eickenberg, A. Bietti, M. Cranmer, G. Krawezik, F. Lanusse, M. McCabe, R. Ohana, L. Parker, B. R.-S. Blancard, T. Tesileanu, K. Cho, and S. Ho. xVal: A Continuous Numerical Tokenization for Scientific Language Models, Dec. 2024. URL http://arxiv.org/abs/2310.02989. arXiv:2310.02989 [stat].

A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, A. Yang, A. Fan, A. Goyal, A. Hartshorn, A. Yang, A. Mitra, A. Sravankumar, A. Korenev, A. Hinsvark, A. Rao, A. Zhang, A. Rodriguez, A. Gregerson, A. Spataru, B. Roziere, B. Biron, B. Tang, B. Chern, C. Caucheteux, C. Nayak, C. Bi, C. Marra, C. McConnell, C. Keller, C. Touret, C. Wu, C. Wong, C. C. Ferrer, C. Nikolaidis, D. Allonsius, D. Song, D. Pintz, D. Livshits, D. Wyatt, D. Esiobu, D. Choudhary, D. Mahajan, D. Garcia-Olano, D. Perino, D. Hupkes, E. Lakomkin, E. AlBadawy, E. Lobanova, E. Dinan, E. M. Smith, F. Radenovic, F. Guzmán, F. Zhang, G. Synnaeve, G. Lee, G. L. Anderson, G. Thattai, G. Nail, G. Mialon, G. Pang, G. Cucurell, H. Nguyen, H. Korevaar, H. Xu, H. Touvron, I. Zarov, I. A. Ibarra, I. Kloumann, I. Misra, I. Evtimov, J. Zhang, J. Copet, J. Lee, J. Geffert, J. Vranes, J. Park, J. Mahadeokar, J. Shah, J. v. d. Linde, J. Billock, J. Hong, J. Lee, J. Fu, J. Chi, J. Huang, J. Liu, J. Wang, J. Yu, J. Bitton, J. Spisak, J. Park, J. Rocca, J. Johnstun, J. Saxe, J. Jia, K. V. Alwala, K. Prasad, K. Upasani, K. Plawiak, K. Li, K. Heafield, K. Stone, K. El-Arini, K. Iyer, K. Malik, K. Chiu, K. Bhalla, K. Lakhotia, L. Rantala-Yeary, L. v. d. Maaten, L. Chen, L. Tan, L. Jenkins, L. Martin, L. Madaan, L. Malo, L. Blecher, L. Landzaat, L. d. Oliveira, M. Muzzi, M. Pasupuleti, M. Singh, M. Paluri, M. Kardas, M. Tsimpoukelli, M. Oldham, M. Rita, M. Pavlova, M. Kambadur, M. Lewis, M. Si, M. K. Singh, M. Hassan, N. Goyal, N. Torabi, N. Bashlykov, N. Bogoychev, N. Chatterji, N. Zhang, O. Duchenne, O. Çelebi, P. Alrassy, P. Zhang, P. Li, P. Vasic, P. Weng, P. Bhargava, P. Dubal, P. Krishnan, P. S. Koura, P. Xu, Q. He, Q. Dong, R. Srinivasan, R. Ganapathy, R. Calderer, R. S. Cabral, R. Stojnic, R. Raileanu, R. Maheswari, R. Girdhar, R. Patel, R. Sauvestre, R. Polidoro, R. Sumbaly, R. Taylor, R. Silva, R. Hou, R. Wang, S. Hosseini, S. Chennabasappa, S. Singh, S. Bell, S. S. Kim, S. Edunov, S. Nie, S. Narang, S. Raparthy, S. Shen, S. Wan, S. Bhosale, S. Zhang, S. Vandenhende, S. Batra, S. Whitman, S. Sootla, S. Collot, S. Gururangan, S. Borodinsky, T. Herman, T. Fowler, T. Sheasha, T. Georgiou, T. Scialom, T. Speckbacher, T. Mihaylov, T. Xiao, U. Karn, V. Goswami, V. Gupta, V. Ramanathan, V. Kerkez, V. Gonguet, V. Do, V. Vogeti, V. Albiero, V. Petrovic, W. Chu, W. Xiong, W. Fu, W. Meers, X. Martinet, X. Wang, X. Wang, X. E. Tan, X. Xia, X. Xie, X. Jia,

X. Wang, Y. Goldschlag, Y. Gaur, Y. Babaei, Y. Wen, Y. Song, Y. Zhang, Y. Li, Y. Mao, Z. D. Coudert, Z. Yan, Z. Chen, Z. Papakipos, A. Singh, A. Srivastava, A. Jain, A. Kelsey, A. Shajnfeld, A. Gangidi, A. Victoria, A. Goldstand, A. Menon, A. Sharma, A. Boesenberg, A. Baevski, A. Feinstein, A. Kallet, A. Sangani, A. Teo, A. Yunus, A. Lupu, A. Alvarado, A. Caples, A. Gu, A. Ho, A. Poulton, A. Ryan, A. Ramchandani, A. Dong, A. Franco, A. Goyal, A. Saraf, A. Chowdhury, A. Gabriel, A. Bharambe, A. Eisenman, A. Yazdan, B. James, B. Maurer, B. Leonhardi, B. Huang, B. Loyd, B. D. Paola, B. Paranjape, B. Liu, B. Wu, B. Ni, B. Hancock, B. Wasti, B. Spence, B. Stojkovic, B. Gamido, B. Montalvo, C. Parker, C. Burton, C. Mejia, C. Liu, C. Wang, C. Kim, C. Zhou, C. Hu, C.-H. Chu, C. Cai, C. Tindal, C. Feichtenhofer, C. Gao, D. Civin, D. Beaty, D. Kreymer, D. Li, D. Adkins, D. Xu, D. Testuggine, D. David, D. Parikh, D. Liskovich, D. Foss, D. Wang, D. Le, D. Holland, E. Dowling, E. Jamil, E. Montgomery, E. Presani, E. Hahn, E. Wood, E.-T. Le, E. Brinkman, E. Arcaute, E. Dunbar, E. Smothers, F. Sun, F. Kreuk, F. Tian, F. Kokkinos, F. Ozgenel, F. Caggioni, F. Kanayet, F. Seide, G. M. Florez, G. Schwarz, G. Badeer, G. Swee, G. Halpern, G. Herman, G. Sizov, Guangyi, Zhang, G. Lakshminarayanan, H. Inan, H. Shojanazeri, H. Zou, H. Wang, H. Zha, H. Habeeb, H. Rudolph, H. Suk, H. Aspegren, H. Goldman, H. Zhan, I. Damlaj, I. Molybog, I. Tufanov, I. Leontiadis, I.-E. Veliche, I. Gat, J. Weissman, J. Geboski, J. Kohli, J. Lam, J. Asher, J.-B. Gaya, J. Marcus, J. Tang, J. Chan, J. Zhen, J. Reizenstein, J. Teboul, J. Zhong, J. Jin, J. Yang, J. Cummings, J. Carvill, J. Shepard, J. McPhie, J. Torres, J. Ginsburg, J. Wang, K. Wu, K. H. U, K. Saxena, K. Khandelwal, K. Zand, K. Matosich, K. Veeraraghavan, K. Michelena, K. Li, K. Jagadeesh, K. Huang, K. Chawla, K. Huang, L. Chen, L. Garg, L. A, L. Silva, L. Bell, L. Zhang, L. Guo, L. Yu, L. Moshkovich, L. Wehrstedt, M. Khabsa, M. Avalani, M. Bhatt, M. Mankus, M. Hasson, M. Lennie, M. Reso, M. Groshev, M. Naumov, M. Lathi, M. Keneally, M. Liu, M. L. Seltzer, M. Valko, M. Restrepo, M. Patel, M. Vyatskov, M. Samvelyan, M. Clark, M. Macey, M. Wang, M. J. Hermoso, M. Metanat, M. Rastegari, M. Bansal, N. Santhanam, N. Parks, N. White, N. Bawa, N. Singhal, N. Egebo, N. Usunier, N. Mehta, N. P. Laptev, N. Dong, N. Cheng, O. Chernoguz, O. Hart, O. Salpekar, O. Kalinli, P. Kent, P. Parekh, P. Saab, P. Balaji, P. Rittner, P. Bontrager, P. Roux, P. Dollar, P. Zvyagina, P. Ratanchandani, P. Yuvraj, Q. Liang, R. Alao, R. Rodriguez, R. Ayub, R. Murthy, R. Nayani, R. Mitra, R. Parthasarathy, R. Li, R. Hogan, R. Battey, R. Wang, R. Howes, R. Rinott, S. Mehta, S. Siby, S. J. Bondu, S. Datta, S. Chugh, S. Hunt, S. Dhillon, S. Sidorov, S. Pan, S. Mahajan, S. Verma, S. Yamamoto, S. Ramaswamy, S. Lindsay, S. Lindsay, S. Feng, S. Lin, S. C. Zha, S. Patil, S. Shankar, S. Zhang, S. Zhang, S. Wang, S. Agarwal, S. Sajuyigbe, S. Chintala, S. Max, S. Chen, S. Kehoe, S. Satterfield, S. Govindaprasad, S. Gupta, S. Deng, S. Cho, S. Virk, S. Subramanian, S. Choudhury, S. Goldman, T. Remez, T. Glaser, T. Best, T. Koehler, T. Robinson, T. Li, T. Zhang, T. Matthews, T. Chou, T. Shaked, V. Vontimitta, V. Ajayi, V. Montanez, V. Mohan, V. S. Kumar, V. Mangla, V. Ionescu, V. Poenaru, V. T. Mihailescu, V. Ivanov, W. Li, W. Wang, W. Jiang, W. Bouaziz, W. Constable, X. Tang, X. Wu, X. Wang, X. Wu, X. Gao, Y. Kleinman, Y. Chen, Y. Hu, Y. Jia, Y. Qi, Y. Li, Y. Zhang, Y. Zhang, Y. Adi, Y. Nam, Yu, Wang, Y. Zhao, Y. Hao, Y. Qian, Y. Li, Y. He, Z. Rait, Z. DeVito, Z. Rosnbrick, Z. Wen, Z. Yang, Z. Zhao, and Z. Ma. The Llama 3 Herd of Models, Nov. 2024. URL http://arxiv.org/abs/2407.21783. arXiv:2407.21783 [cs].

N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson. Large Language Models Are Zero-Shot Time Series Forecasters, Aug. 2024. URL http://arxiv.org/abs/2310.07820. arXiv:2310.07820 [cs].

S. Hegselmann, A. Buendia, H. Lang, M. Agrawal, X. Jiang, and D. Sontag. TabLLM: Few-shot Classification of Tabular Data with Large Language Models, Mar. 2023. URL http://arxiv.org/abs/2210.10723. arXiv:2210.10723 [cs].

J. Herzen, F. Lässig, S. G. Piazzetta, T. Neuer, L. Tafti, G. Raille, T. V. Pottelbergh, M. Pasieka, A. Skrodzki, N. Huguenin, M. Dumonal, J. Kościsz, D. Bader, F. Gusset, M. Benheddi, C. Williamson, M. Kosinski, M. Petrik, and G. Grosch. Darts: User-Friendly Modern Machine Learning for Time Series. *Journal of Machine Learning Research*, 23(124):1–6, 2022. ISSN 1533-7928. URL http://jmlr.org/papers/v23/21-1177.html.

R. Koenker and K. F. Hallock. Quantile Regression. *Journal of Economic Perspectives*, 15(4): 143–156, Dec. 2001. ISSN 0895-3309. doi: 10.1257/jep.15.4.143. URL https://www.aeaweb.org/articles?id=10.1257/jep.15.4.143.

B. Koloski, A. Margeloiu, X. Jiang, B. Škrlj, N. Simidjievski, and M. Jamnik. LLM Embeddings for Deep Learning on Tabular Data, Feb. 2025. URL http://arxiv.org/abs/2502.11596. arXiv:2502.11596 [cs] version: 1.

J. Lindsey, W. Gurnee, E. Ameisen, B. Chen, A. Pearce, N. L. Turner, C. Citro, D. Abrahams, S. Carter, B. Hosmer, J. Marcus, M. Sklar, A. Templeton, T. Bricken, C. McDougall, H. Cunningham, T. Henighan, A. Jermyn, A. Jones, A. Persic, Z. Qi, T. B. Thompson, S. Zimmerman, K. Rivoire, T. Conerly, C. Olah, and J. Batson. On the Biology of a Large Language Model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

J. Requeima, J. Bronskill, D. Choi, R. E. Turner, and D. Duvenaud. LLM Processes: Numerical Predictive Distributions Conditioned on Natural Language, Dec. 2024. URL http://arxiv.org/abs/2405.12856. arXiv:2405.12856 [stat].

E. Schwartz, L. Choshen, J. Shtok, S. Doveh, L. Karlinsky, and A. Arbelle. NumeroLogic: Number Encoding for Enhanced LLMs' Numerical Reasoning, Mar. 2024. URL http://arxiv.org/abs/2404.00459. arXiv:2404.00459 [cs] version: 1.

A. Shysheya, J. Bronskill, J. Requeima, S. A. Siddiqui, J. Gonzalez, D. Duvenaud, and R. E. Turner. JoLT: Joint Probabilistic Predictions on Tabular Data Using LLMs, Feb. 2025. URL http://arxiv.org/abs/2502.11877. arXiv:2502.11877 [stat].

A. K. Singh and D. J. Strouse. Tokenization counts: the impact of tokenization on arithmetic in frontier LLMs, Feb. 2024. URL http://arxiv.org/abs/2402.14903. arXiv:2402.14903 [cs].

A. Stolfo, Y. Belinkov, and M. Sachan. A Mechanistic Interpretation of Arithmetic Reasoning in Language Models using Causal Mediation Analysis, Oct. 2023. URL http://arxiv.org/abs/2305.15054. arXiv:2305.15054 [cs].

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, July 2023. URL http://arxiv.org/abs/2307.09288. arXiv:2307.09288 [cs].

R. Vacareanu, V.-A. Negru, V. Suciu, and M. Surdeanu. From Words to Numbers: Your Large Language Model Is Secretly A Capable Regressor When Given In-Context Examples, Sept. 2024. URL http://arxiv.org/abs/2404.07544. arXiv:2404.07544 [cs].

E. Wallace, Y. Wang, S. Li, S. Singh, and M. Gardner. Do NLP Models Know Numbers? Probing Numeracy in Embeddings, Sept. 2019. URL http://arxiv.org/abs/1909.07940. arXiv:1909.07940 [cs].

H. Xue and F. D. Salim. PromptCast: A New Prompt-based Learning Paradigm for Time Series Forecasting, Dec. 2023. URL http://arxiv.org/abs/2210.08964. arXiv:2210.08964 [stat].

S. Yang, L. Wang, and P. Ding. Causal inference with confounders missing not at random, Feb. 2019. URL http://arxiv.org/abs/1702.03951. arXiv:1702.03951 [stat].

Y. Zhou, U. Alon, X. Chen, X. Wang, R. Agarwal, and D. Zhou. Transformers Can Achieve Length Generalization But Not Robustly, Feb. 2024. URL http://arxiv.org/abs/2402.09371. arXiv:2402.09371 [cs].

A. Y. Zhu, N. Mitra, and J. Roy. Addressing positivity violations in causal effect estimation using Gaussian process priors. *Statistics in Medicine*, 42(1):33–51, 2023. ISSN 1097-0258. doi: 10.1002/sim.9600. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/sim.9600. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sim.9600.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claim that our results suggest that LLM embeddings carry informative signals about summary statistics of their predictive distributions is justified in Section 3. The claim that our results suggest that LLM embeddings carry informative signals about numerical uncertainty is justified by experimental results in Section 4. The empirical results that justify that our results might lead to reduced computational overhead are presented in Figure 11.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper focuses on experimental results, with no novel theoretical results introduced.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Most important details of the experiments are presented in Sections 3-5, in a dedicated *Method of Investigation* section. Further details are provided in the Appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code to reproduce the experiments can be found at https://anonymous.4open.science/r/guess_llm-811B/.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Details of the data splits, hyperparameters, and optimization process can be found in the provided code, and are also described in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Where relevant, the plots include the necessary error bars (e.g. Figure 11, Figure 6).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The information about the computer resources used can be found in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The paper does not involve human subjects or participants. No new real-world datasets are released as a part of this project. The project does not raise any significant ethical considerations listed in the 'Societal Impact' section.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The paper provides an investigation into the numerical predictive distributions of the LLMs, and whether they can be approximated using LLM's internal representations and as such, it does not have a significant societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not release new datasets or pre-trained models.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Any existing assets are listed and credited in the Appendix.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The research does not involve human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The relevant description of the way that the LLMs are used in the experiments presented in this paper is included in the Appendix as well as in the provided code.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

A Details of the Experimental Setup

A.0.1 Assets and Licensing Information

The following existing assets were used to produce the experimental results:

- **Monash dataset** Godahewa et al. [2021]
- **Darts dataset** Herzen et al. [2022]
- **Llama-2-7B model** Touvron et al. [2023]
- **Llama-3-8B model** Grattafiori et al. [2024]
- **Phi-3.5-mini model** Abdin et al. [2024]

A.1 Computer infrastructure used

Hardware. All experiments were conducted using 2 separate NC24rs_v3 instances and one NC80adis_H100_v5 instance on the Microsoft Azure cloud platform. This instances are a part of Azure's GPU-optimised virtual machine series, with their hardware specifications summarised in Table 3.

Table 3: Azure Virtual Machine Specifications

Specification	NC24rs_v3	NC80adis_H100_v5
vCPUs	24	80
System Memory (GiB)	448	640
GPU Model	4× NVIDIA Tesla V100	2× NVIDIA H100 NVL
GPU Memory (per GPU)	16 GiB	94 GiB
Total GPU Memory	64 GiB	188 GiB
GPU Architecture	Volta	Hopper
CUDA Version	11.x	12.x
CPU Model	Intel Xeon E5-2690 v4	AMD EPYC Genoa
Local Storage	2.9 TB	7.1 TB

Generating the synthetic dataset for one scaling factor $\ell \in \{1, 10, 1000, 10000\}$ took no more than 10h. Training one probe model took no more than 4h.

A.2 Details of the datasets

A.2.1 Details of the synthetic time series dataset

We generate a synthetic dataset comprising time series derived from a family of parametric functions, each evaluated over a fixed domain and perturbed with controlled noise. The purpose is to simulate diverse temporal patterns, inducing varying levels of uncertainty in the LLM's predictions.

We use a set of base functions defined over the interval $x \in [0, 60]$, discretized into 120 equidistant points. The functions are summarised in Table 4. For each function and value of a, we generate a clean series $y = f(a \cdot x)$, and then apply:

- Additive Gaussian noise with variance $\sigma^2 \in \{0.0, 0.01, 0.05, 0.1\}$.
- Vertical scaling by $b \sim \mathcal{U}(0, \ell)$
- Vertical translation by $d \sim \mathcal{U}(-\ell, \ell)$

From each transformed series, we sample 10 different subsequences for each of the lengths $n \in \{3, 5, 7, 10, 13, 15, 17, 20, 25, 30, 35, 40\}$, with each subsequence starting at a random offset. Each sequence becomes a training input. Inputs are serialized as floating-point strings with a user-defined number of decimal places p (we use $p = 4$ for $\ell = 1.0$, $p = 3$ for $\ell = 10.0$, $p = 2$ for $\ell = 1000.0$ and $p = 1$ for $\ell = 10000.0$). This results in 33600 generated time series for each value of ℓ.

Concatenated dataset. Having constructed the individual dataset for each scaling factor $\ell \in \{1, 10, 1000, 10000\}$, we also construct one concatenated dataset. In doing that, we limit the number of datapoints to 80000 and ensure that the y_{test} values of the generated time series are equally distributed on the log scale, from 10^{-2} to 10^{4}. This is to ensure a balanced distribution of the train and test examples.

Dataset filtering. Before using the generated datasets for training the probing models, we apply dataset filtering to exclude any potential outliers. Namely, we ensure that the mean, median and greedy LLM prediction lie in $[-\ell, \ell]$.

Function name	Formula	a-range
`sin`	$\sin(x)$	[0.5, 6.0]
`linear_sin`	$0.2 \cdot \sin(x) + \frac{x}{450}$	[0.5, 6.0]
`sinc`	$\mathrm{sinc}(x)$	[0.05, 0.2]
`xsine`	$\frac{x-30}{50} \cdot \sin(x-30)$	[0.5, 1.3]
`beat`	$\sin(x) \cdot \sin\left(\frac{x}{2}\right)$	[0.1, 6.0]
`gaussian_wave`	$e^{-\frac{(x-2)^2}{2}} \cdot \cos(10\pi(x-2))$	[0.01, 0.1]
`random`	$\mathcal{U}(-1, 1)$	[0.0, 1.0]

Table 4: Functions used to generate time series data, their mathematical forms, and the range of the time-scaling parameter a.

A.2.2 Monash dataset

- **Data Loading**: We use the data from the Monash dataset, preprocessed by Gruver et al. [2024] and available from https://drive.google.com/file/d/1sKrpWbD3LvLQ_e5lWgX3wJqT50sTd1aZ/view?usp=sharing. Each sub-dataset file contains tuples of the form `(train,test)`, which are concatenated to form full univariate time series.
- **Resampling**: To ensure computational tractability, each series is subsampled (via strided slicing) to contain at most 1000 time steps.
- **Series Selection**: For each dataset, a maximum of 50 time series are selected at random to control the number of examples used during training.
- **Subsequence Generation**: From each selected series, we extract multiple training subsequences of varying lengths $n \in \{3, 5, 7, 10, 13, 15, 17, 20, 25, 30, 35, 40\}$. For each length, we generate up to 10 training subsequences, sampled at different offsets.

A.2.3 Darts dataset

- **Data Loading**: We use the data from the Darts dataset, available from the `darts` python package. We use the following sub-datasets: AirPassengersDataset, AusBeerDataset, GasRateCO2Dataset, MonthlyMilkDataset, SunspotsDataset, WineDataset, WoolyDataset, HeartRateDataset.
- **Resampling**: To ensure computational tractability, the series for the datasets SunspotsDataset and HeartRateDataset are subsampled (via strided slicing).
- **Series Selection**: For each dataset, all available time series are selected.
- **Subsequence Generation**: From each selected series, we extract multiple training subsequences of varying lengths $n \in \{3, 5, 7, 10, 13, 15, 17, 20, 25, 30, 35, 40\}$. For each length, we generate up to 10 training subsequences, sampled at different offsets.

A.2.4 LLM generation settings

We generate the LLM hidden states from a Llama-2-7B model, available through the `huggingface` library. Each of the generated time series, we obtain 100 samples from the LLM, generated autoregressively, as well as the greedy generation. As the Llama-2 tokenizer encodes each digit separately, during generation we narrow down the generated tokens to digits, decimal point and $+/-$ signs. For obtaining the random samples, we use `temperature=1.0` and `top_p=0.95`. We exclude from the final dataset samples for which generation failed at least once (i.e. the obtained generation was not a valid number), such that each time series in the final dataset has exactly 100 LLM samples.

A.2.5 Train-validation-test split

Before training, we split each of the datasets in 80% training dataset, 10% validation dataset and 10% test dataset. Unless otherwise stated (in the generalisation experiments), these splits are random. We do not apply any scaling or transformation to either the LLM embeddings (which are inputs to our model) or the outputs.

A.3 Details of the magnitude-factorised regression model

Our magnitude-factorised regression models, used both for the purpose of point prediction and for the purpose of quantile regression, has the following hyperparameters. We report the default values of the hyperparameters used in Table 5 and Table 6, and then report any deviations from these values for specific experiments below. We train the model using the ADAM optimiser. For a detailed implementation of the magnitude-factorised regression models, see the provided code.

Hyperparameter	Description	Default Value
`min_mag`	Minimum exponent for base-10 magnitude scaling (as used by f_{order})	-3
`max_mag`	Maximum exponent for base-10 magnitude scaling	$log_{10}\ell$
`use_arctan`	Apply $10 \cdot \arctan(0.5 \cdot x)$ to bound output of f_{val}	`True`
`beta`	Weight for regression loss component	10.0
`K`	Top-K exponents taken into consideration (see Equation 4)	3
`hidden_layers`	Number of hidden layers in feature extractor	1
`hidden_dim`	Dimensionality of hidden feature representation	512
`hidden_states_list`	A list of the hidden states $\mathcal{H}$ to use as input	$[25, \ldots, 32]$
`quantile_weights`	Weights of each of the quantiles in the quantile regression loss function	$[1, 1, 2, 5, 2, 1, 1]$

Table 5: Model-specific hyperparameters for the magnitude-factorised regression model.

Hyperparameter	Description	Default Value
`learning_rate`	Learning rate for the optimizer	10^{-4}
`weight_decay`	L2 regularization weight	0.1
`scheduler_step_size`	Learning rate scheduler step size	100
`scheduler_gamma`	Learning rate scheduler step size	0.5
`batch_size`	Number of samples per training batch	1024
`max_epochs`	Number of training epochs	500
`patience`	Patience for the early stopping	200

Table 6: Optimizer and training-related hyperparameters.

A.3.1 Experiment-specific hyperparameter settings

Figure 2. We use `max_mag` $= 4$.

Figure 3 and Figure 1. We use `lr` $= 10^{-4}$, `max_epochs` $= 2000$.

Figure 4 and Table 1. We use `max_mag` $= 13$.

Figure 7. We use `batch_size` $= 2048$, `lr` $= 10^{-5}$ and `max_mag` $= 13$.

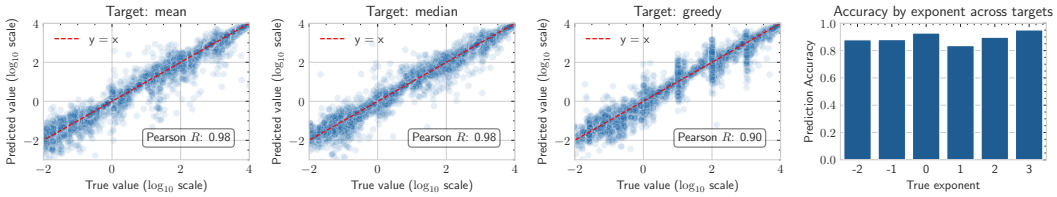
B Additional Experimental Results

B.1 Results with other LLMs

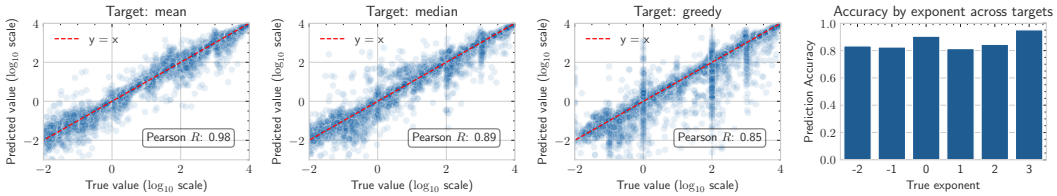
We provide results for the key experiments in the main paper with two other LLMs: Llama-3-8B and Phi-3.5-mini-instruct. As the tokenizers of these models do not encode digits separately, we *do not* narrow down the generated tokens during decoding. For obtaining the random samples, we use `temperature=1.0` and `top_p=0.95`. We perform repeated sampling until for each time series, we obtain 100 LLM samples $y_i^j \sim p_{\text{LLM}}(\cdot|\mathbf{x}_i)$.

B.2 Sample Efficiency Results

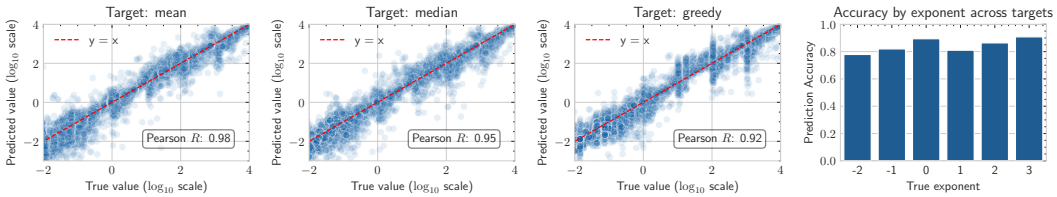
In fig 11 we provide sample efficiency results across all dataset scales and across the three statistics: the median, the first quartile (Q1) and the third quartile (Q3). Results are presented for the quantile regression model as described in section 4.



(a) Llama-2-7B



(b) Llama-3-8B



(c) Phi-3.5-instruct

Figure 8: Predicted vs. sample mean, median and greedy prediction (on $\log_{10}$ scale).

Table 7: MSE for the predictions on the dataset with scale $\ell = 1.0$, reported for all models.

(a) Llama-2-7B

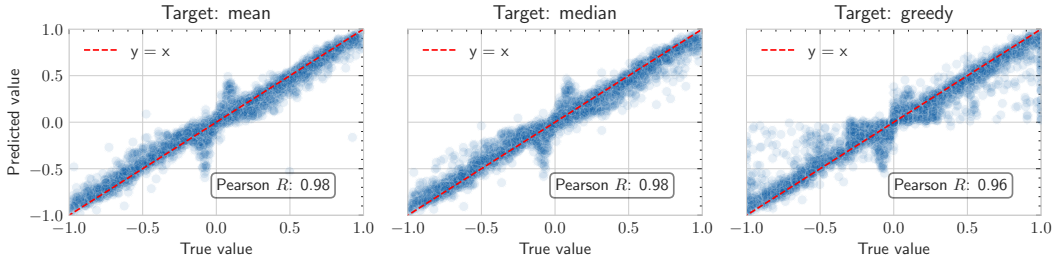
target	$\hat{y}$ (ours)	$\bar{\mathbf{x}}$	$\bar{\mathbf{x}}_i$	$x_{i,n}$
mean	0.009	0.256	0.035	0.085
median	0.009	0.260	0.041	0.087
greedy	0.024	0.273	0.065	0.109

(b) Llama-3-8B

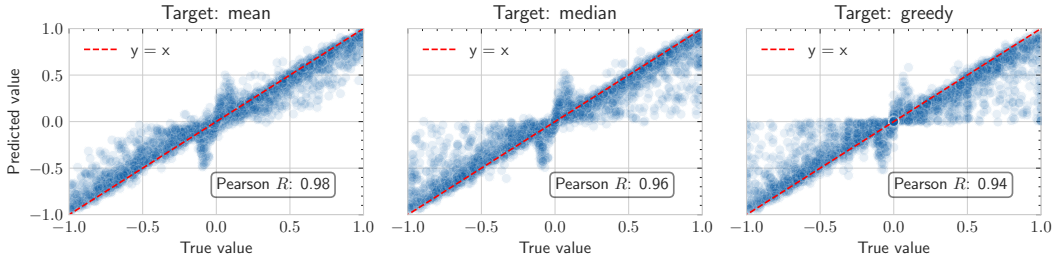
target	$\hat{y}$ (ours)	$\bar{\mathbf{x}}$	$\bar{\mathbf{x}}_i$	$x_{i,n}$
mean	0.014	0.253	0.047	0.093
median	0.025	0.264	0.061	0.106
greedy	0.033	0.255	0.072	0.122

(c) Phi-3.5-mini-instruct

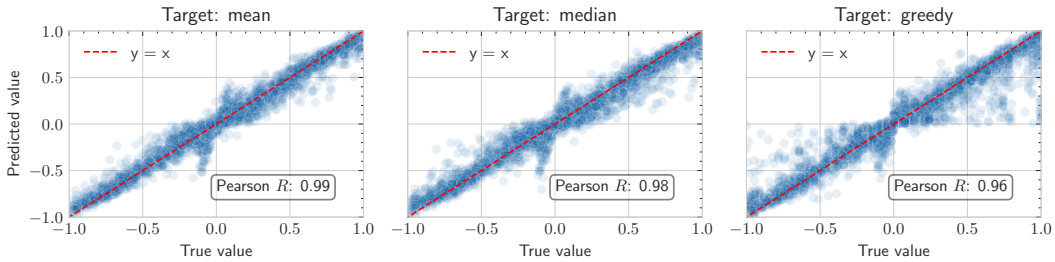
target	$\hat{y}$ (ours)	$\bar{\mathbf{x}}$	$\bar{\mathbf{x}}_i$	$x_{i,n}$
mean	0.007	0.248	0.042	0.100
median	0.010	0.252	0.047	0.104
greedy	0.021	0.270	0.060	0.113



(a) Llama-2-7B



(b) Llama-3-8B



(c) Phi-3.5-instruct

Figure 9: Predicted vs. sample mean, median and greedy prediction on the dataset with scale $\ell = 1.0$.

Table 8: Coverage of the CI for all models.

(a) Llama-2-7B

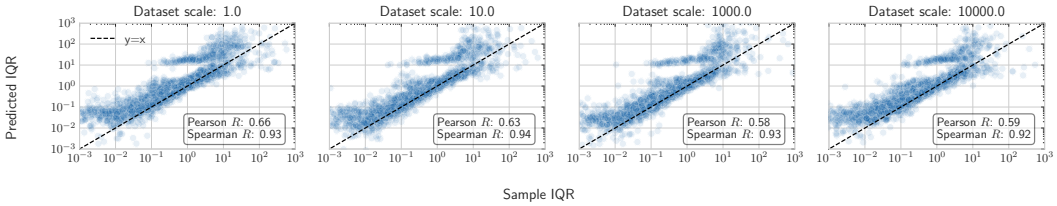
α dataset	50%	90%	95%
1.0	47.3 ± 0.6	88.2 ± 0.4	93.3 ± 0.3
10.0	52.3 ± 0.6	89.5 ± 0.4	93.8 ± 0.3
1000.0	48.9 ± 0.6	87.2 ± 0.4	92.7 ± 0.3
10000.0	46.5 ± 0.6	86.0 ± 0.5	91.3 ± 0.4

(b) Llama-3-8B

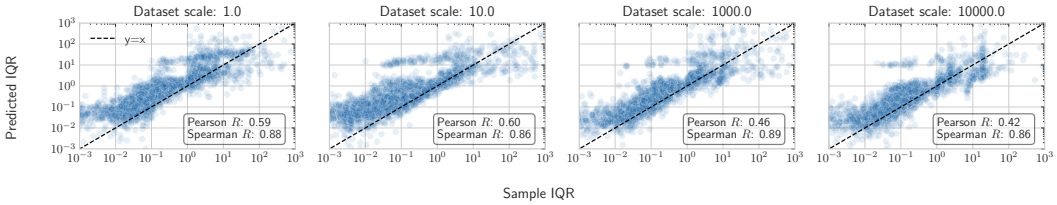
α dataset	50%	90%	95%
1.0	48.1 ± 0.6	88.2 ± 0.4	93.4 ± 0.3
10.0	50.6 ± 0.6	89.3 ± 0.4	93.9 ± 0.3
1000.0	48.9 ± 0.6	87.2 ± 0.4	92.7 ± 0.3
10000.0	45.5 ± 0.6	85.7 ± 0.5	91.2 ± 0.4

(c) Phi-3.5-mini-instruct

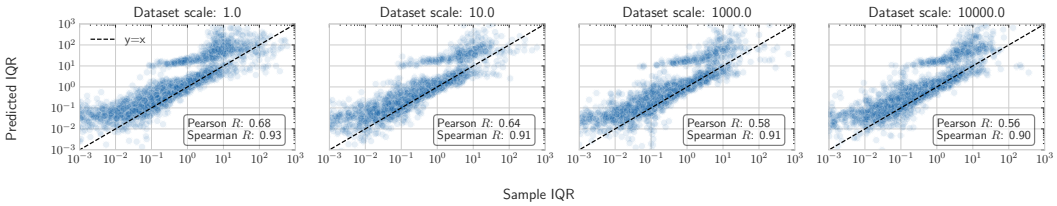
α dataset	50%	90%	95%
1.0	51.1 ± 0.5	89.5 ± 0.4	94.7 ± 0.3
10.0	49.0 ± 0.5	89.1 ± 0.4	93.6 ± 0.3
1000.0	49.1 ± 0.5	88.5 ± 0.4	93.0 ± 0.3
10000.0	49.2 ± 0.5	87.6 ± 0.4	93.0 ± 0.3



(a) Llama-2-7B

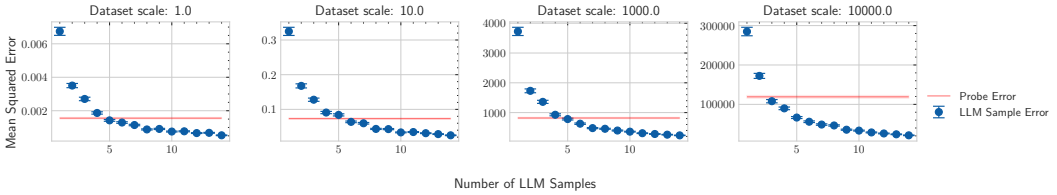


(b) Llama-3-8B

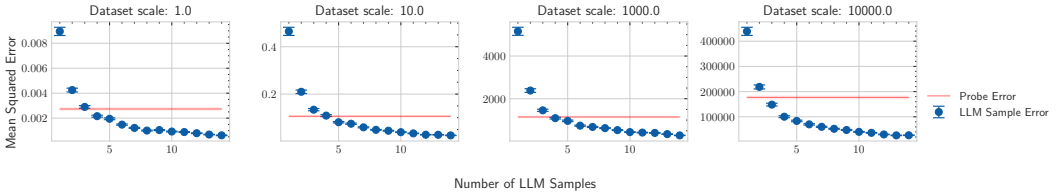


(c) Phi-3.5-instruct

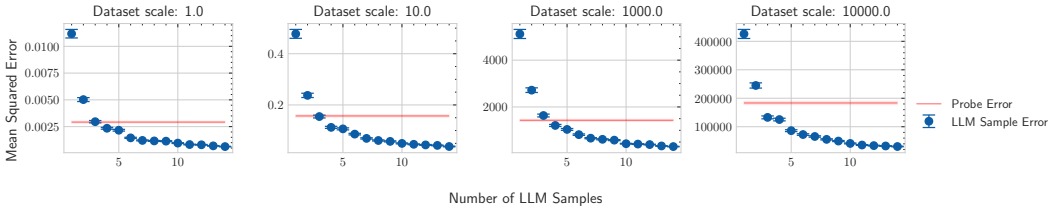
Figure 10: Predicted IQR vs. Sample IQR (median adjusted).



(a) Median



(b) Q1



(c) Q3

Figure 11: Sample efficiency of estimating the median, the first (Q1) and the third (Q3) quartiles.