PREDICTING THE PERFORMANCE OF BLACK-BOX LLMS THROUGH SELF-QUERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

As large language models (LLMs) are increasingly relied on in AI systems, predicting when they make mistakes is crucial. While a great deal of work in the field uses internal representations to interpret model behavior, these representations are inaccessible when given solely black-box access through an API. In this paper, we extract features of LLMs in a black-box manner by asking simple elicitation questions and use the probabilities of different responses *as* features to train reliable predictors of model behavior. We demonstrate that training a linear model on these low-dimensional features produces reliable and generalizable predictors of model performance at the instance level (e.g., if a particular generation correctly answers a question). Remarkably, these can often outperform white-box linear predictors that operate over a model's hidden state or the full distribution over its vocabulary. In addition, we demonstrate that these extracted features can be used to evaluate more nuanced aspects of a language model's state. For instance, they can be used to distinguish between GPT-3.5 and a version of GPT-3.5 affected by an adversarial system prompt that makes its answers often incorrect. Furthermore, they can reliably distinguish between different model architectures and sizes, enabling the detection of misrepresented models provided through an API (e.g., identifying if GPT-3.5 is supplied instead of GPT-4).

1 INTRODUCTION

Large language models (LLMs) have demonstrated strong performance on a wide variety of tasks (Radford et al.), leading to their increased involvement in larger systems. For instance, they are often used to provide supervision (Bai et al., 2022) or as tools in decision-making (Benary et al., 2023; Sha et al., 2023). Thus, it is crucial to understand and predict their behaviors, especially in high-stakes settings. However, as with any deep network, it is difficult to understand the behavior of such large models (Zhang et al., 2021). For instance, prior work has studied input gradients or saliency maps (Simonyan et al., 2013; Zeiler & Fergus, 2014; Pukdee et al., 2024)) to attempt to understand neural network behavior, but this can fail to reliably describe model behavior (Adebayo et al., 2018; Kindermans et al., 2019; Srinivas & Fleuret, 2020). Other work has studied the ability of transformers to represent certain algorithms (Nanda et al., 2022; Zhong et al., 2024) that may be involved in their predictions.

One promising direction in understanding LLMs (or any other multimodal model that understands natural language) is to leverage their ability to interact with human queries. Recent work has demonstrated that a LLM's hidden state contains low-dimensional representations of model truthfulness or harmfulness (Zou et al., 2023a). Other work studies learning sparse dictionaries and analyzing how these networks activate on certain, related input tokens (Bricken et al., 2023). While significant progress has been made on these fronts, these approaches all require white-box access to these models (i.e., access to the model's activations or hidden states). However, many of the best-performing LLMs (Achiam et al., 2023; Team et al., 2023) lie beyond closed-source APIs, so these prior attempts to understand model behavior cannot be applied. This raises the question, "*How well can we model the LLM's behavior with only black-box access?*"

In this paper, we propose to extract useful features in predicting model performance from black-box LLMs by eliciting model responses by querying these LLMs about their outputs. In essence, after receiving a generation or prediction from a LLM, we leverage the LLM's ability to reason about its own generated answer and meaningfully respond to follow-up questions, such as, "Are you able

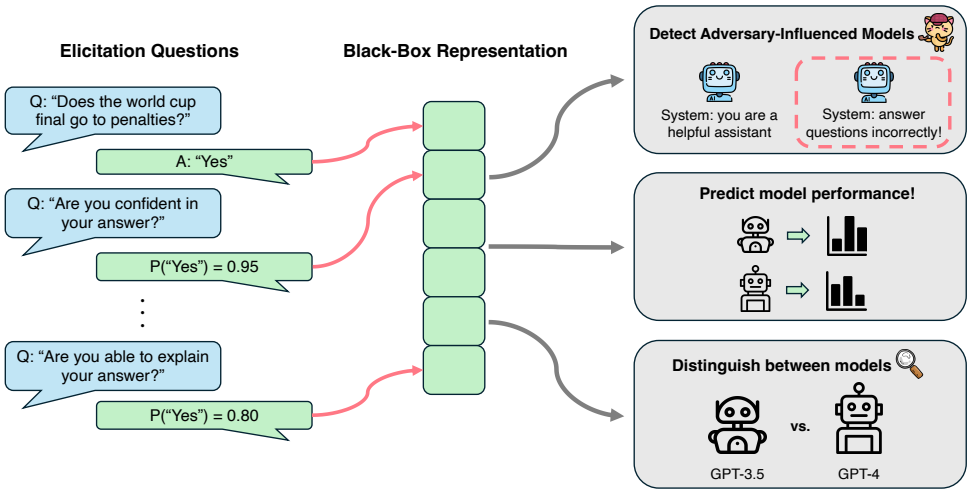

Figure 1: Our approach to extract black-box representations from LLMs which can be used for various applications, including predicting performance, determining if correct models are given through an API, and detecting models that have been influenced by adversarial system prompts.

to explain your answer?" Our hypothesis is that the probability distribution over answers to these questions significantly varies between whether the model's original answer is correct, as well as for different model classes and sizes. As we only look at the outputs of these LLMs (i.e., top-k token probabilities that are accessible through many APIs), we remark that this approach is both model-agnostic and works for closed-source models. When top-k probabilities are not provided, we can approximate this by sampling from the LLM, and we provide a result on how quickly this sampling-based approximation converges to the approach with access to the true underlying probabilities. We remark that while our features are useful for predicting model behavior and other applications, they differ in nature from alternative approaches in mechanistic interpretability (Nanda et al., 2023) or in reasoning via Chain-of-Thought Wei et al. (2022), which try to extract the underlying reasoning behavior of LLMs.

In our experiments, we demonstrate that our proposed approach of querying a model with elicitation questions produces black-box features that are useful in a variety of applications in predicting model performance. We first demonstrate that they can be used to train accurate predictors of model performance (e.g., predicting whether a particular class prediction or text generation is correct). Our approach of querying a model with elicitation questions often matches or outperforms linear predictors that operate over the LLM's hidden state (i.e., requiring white-box access), over a wide variety of LLMs applied to question-answering (QA) tasks. As our extracted representations are low-dimensional, we also demonstrate that predictors trained on them observe stronger generalization guarantees. We also show that sampling-based approximations closely match the performance of a model that uses the true probabilities, so our method performs well even without access to top-k probabilities.

In addition to predicting LLM performance at the example level, these extracted features are also useful for a variety of other applications in assessing the state of a LLM. For instance, recent work demonstrated that model internals can be used to assess when an LLM has been adversarially influenced by a prompt (MacDiarmid et al., 2024) to exhibit harmful behavior. Our work extends this setting by demonstrating that our extracted representations can be used to almost perfectly detect when a LLM (e.g., GPT-3.5) has been adversarially influenced by a system prompt in a *completely black-box* setting. We also provide evidence that our approach is robust to variations in the system prompt. Finally, we also demonstrate that our approach can be used to reliably distinguish between different model architectures and model sizes; this can be useful in evaluating if cheaper or smaller models are falsely being provided through these closed-source APIs.

2 Related Work

Predicting Model Performance Predicting the behavior of deep neural networks is an important problem in the field, due to the difficult-to-interpret nature of these models. Existing work looks to

assess the performance of models by directly operating over the weight space (Unterthiner et al., 2020) or ensembles of multiple trained models (Jiang et al., 2021). Specifically for language models, prior work has primarily focused on predicting task-level performance on new tasks; for instance, developing predictors of task-level performance that use the performance on similar or related tasks (Xia et al., 2020; Ye et al., 2023). Other work attempts to predict the performance of models as they scale up computation (in both terms of data and model size) (Kaplan et al., 2020; Muennighoff et al., 2024). Our work is different as we predict **instance-level performance** (i.e., correctness on a certain input), and we leverage a small amount of labeled data from the downstream task.

Understanding and Benchmarking LLMs A large body of work has focused on understanding the capabilities of LLMs. The field of mechanistic interpretability has recently evolved around understanding the inner workings of LLMs by uncovering circuits or specific weight activations (Olsson et al., 2022; Nanda et al., 2022). This has developed a variety of potential hypotheses for how models learn to perform specific tasks (Zhong et al., 2024), as well as the tendencies of certain activations in a LLM to activate on certain types of inputs (Bills et al., 2023; Sun et al., 2024). Other works have studied model behavior by locating specific regions of a LLM that relate to certain concepts such as untruthfulness (Campbell et al., 2023) or honesty and ethical behavior (Zou et al., 2023a). Our work is different in that we only assume black-box access, with a similar goal to extract information about model behavior. Finally, other work has attempted to study the abilities and performance of LLM via developing challenging benchmarks (Hendrycks et al., 2020), also including those that use techniques from the cognitive sciences (Binz & Schulz, 2023) or by comparing with human similarity judgements (Coda-Forno et al., 2024). While these approaches look to benchmark and quantify performance in aggregate over tasks, our settings looks to predict the performance at the example level, for deciding when to trust or use LLMs in deployment.

Extracting Features from Neural Networks Many other works have explored approaches to extract representations from neural networks (NNs). A related line of work looks to train NNs (specifically image classifiers) to extract a small set of discrete, interpretable concepts, which can be passed through a linear probe to recover a classifier (Koh et al., 2020). In our case, we leverage the ability of the LLM to understand language and can circumvent this need for training, extracting features in a task-agnostic manner. Prior work has studied how to extract useful representations for downstream tasks (Wang et al., 2023; Springer et al., 2024). Our approach significantly differs in nature from these approaches, as we are looking to extract more compressed, low-dimensional features that reveal information about model behavior. Perhaps the most related work employs a similar strategy of asking questions, specifically to detect instances where a model is being untruthful (Pacchiardi et al., 2024). Our work significantly generalizes this approach towards the broader task of predicting model behavior and performance.

Uncertainty Quantification in LLMs Finally, a related notion to our work is assessing the calibration or ability of a language model to represent its own uncertainty (Xiong et al., 2023). Many of the elicitation questions that we ask prompt the model to look at its answer and answer "Yes" or "No"; this is related to the notion of a model's ability to understand what it knows (Kadavath et al., 2022) or reflect uncertainty in its own decisions. Our work is different, however, as we elicit these probabilities as a representation from such a model to train a simple, calibrated linear classifier.

3 Eliciting Black-box Features from Language Models

As we do not assume access to the internals of a LLM, we propose to extract useful features in predicting its behavior by asking eliciting questions about its generations. This approach is completely black-box as we only look at the model's outputs, or more specifically, its top-k probabilities over the next token. We feed these as features into simple linear classifiers for some downstream task (e.g., predicting model performance). We note that for some LLM APIs, we do not have access to top-k probabilities, so we theoretically analyze predictors trained on approximations of these probabilities via sampling.

3.1 Extracting Features by Follow-up Elicitation Questions

To extract our black-box features, we prompt the model with a large number of elicitation questions. We consider a set of questions $Q = \{q_1, ..., q_d\}$ and some autoregressive language

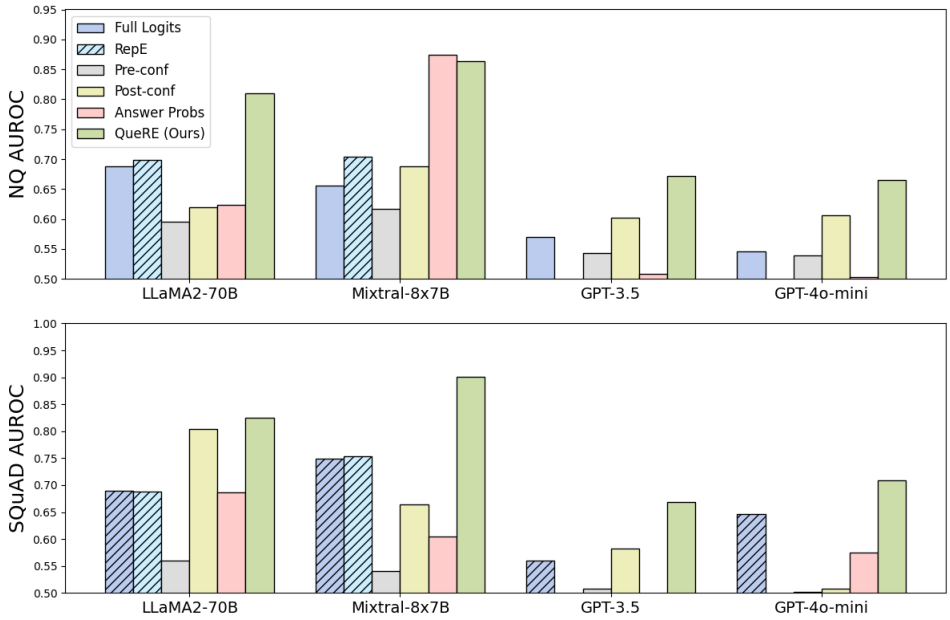

Figure 2: AUROC in predicting model performance on the **open-ended QA benchmarks** of Natural Questions (Top) and SQuAD (Bottom). Dashed bars represent black-box methods, which require more access than our approaches. RepE cannot be applied to black-box models, so it does not have a bar for GPT models. We also note that full logits for the GPT models is an approximation of a sparse vector with nonzero values for the top-5 logits from the API.

model, which models some distribution P over sequences of text. We also consider a dataset $D = \{(x_1, y_1), ..., (x_n, y_n)\}$, where x_i is a sequence of tokens and y_i corresponds to a binary label, for example, if the LLM has correctly answered the question x_i. We define a_i as the greedy response from the LLM, or that $a_i = \arg\max_c P(c|x_i)$. Then, we construct our black-box representation as some vector $z = (z_1, ..., z_d)$, where each $z_j = P(\texttt{yes}|x \oplus a \oplus q_j)$, where $\oplus$ denotes concatenation. In other words, dimensions of our representation correspond to the probability of the `yes` token under the LLM (where the distribution is specified over the `yes` and `no` tokens), in response to the question x, the greedy sampled answer a, and the elicitation question q_j. The elicitation questions are detailed in Appendix J.2, but generally consist of simple self-inquiry questions such as "Do you think your answer is correct?" or "Are your responses free from bias?" This simple approach allows us to add more information to representations by continuing to generate new follow-up questions. In our paper, we find that working with a set of roughly 50 questions seems to be sufficient for strong performance (see ablations in Section 4.5).

In addition to these probabilities of responses to questions, we also append: (1) pre- and post-confidence scores of the LLM, which are responses to asking the question before and after generating a greedy sample from the model, and (2) the distribution over possible answers for the task, (for open-ended QA tasks, we simply use the log probability of the greedy output). In our experiments with GPT-3.5, we also append the sorted top-5 probabilities returned by the API. With these representations, we train a linear predictor β to predict the label y (e.g., whether the model is correct or not) given our representation z.

3.2 Constructing Eliciting Prompts

To construct this set of eliciting questions Q, we specify a small number of questions that relate to the model's confidence or belief in its answer. We also use GPT4 to generate a larger number (40) of questions. The questions and prompts used to generate the GPT4-generated questions are given in Appendix J.2. As noted in prior work that uses similar questions for lie detection (Pacchiardi et al., 2024), a wide variety of questions seems to lead to more useful representations, capturing more information from the LLM.

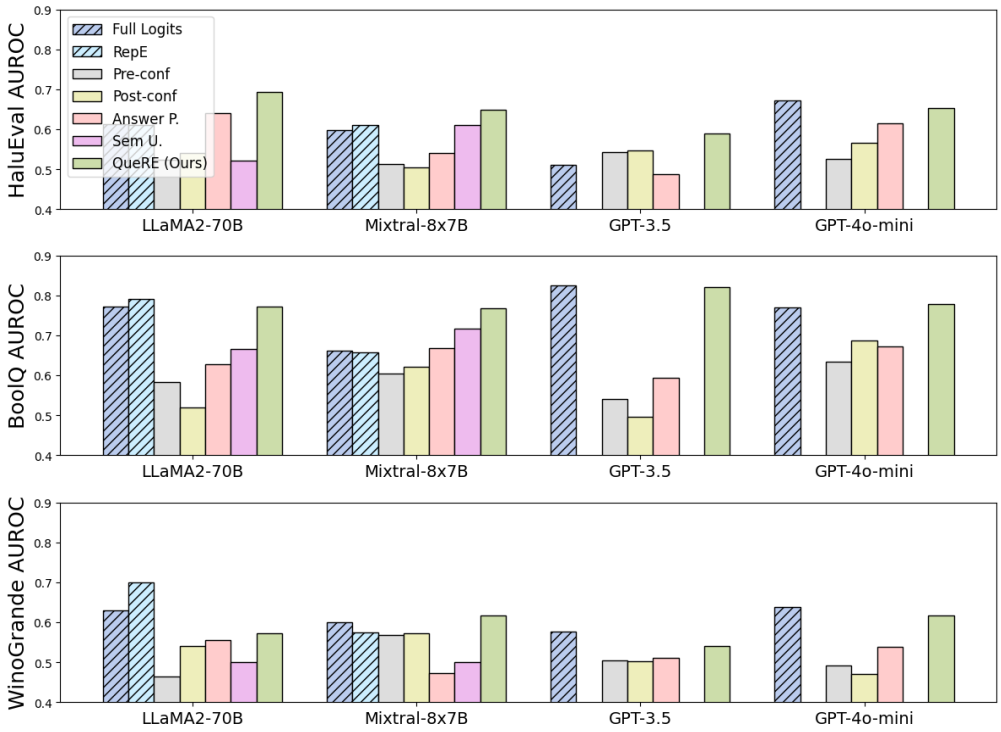

Figure 3: AUROC in predicting model performance on **multiple choice question (MCQ) or true/-false question benchmarks** of HaluEval (Top), BoolQ (Middle), and WinoGrande (Bottom). Dashed bars represent black-box methods. RepE cannot be applied to black-box models, so it does not have a bar for GPT models. Full logits for the GPT models is a sparse vector with nonzero values for the top-5 logits from the API.

We note that based on the specific nature of the question, the response (e.g., the probability of responding `yes`) could define a weak predictor of whether the model is correct or not. This is reminiscent of the design of weak learners in boosting (Freund & Schapire, 1996) or weak labelers in programmatic weak supervision (Ratner et al., 2017; Smith et al., 2024; Sam & Kolter, 2023). However, to maintain our approach's generality and to not restrict our approach to only a certain type of elicitation questions, we treat these as abstract features for a linear predictor. We also note that further work could perform discrete optimization over prompts to further improve the extracted representation's usability, through methods described in (Wen et al., 2024; Zou et al., 2023b; Chao et al., 2023). However, one key appeal of the current approach is that it defines an extremely simple classifier in a task-agnostic fashion. Performing optimization over these questions might lead to overfitting, and the resulting predictors on the outputs of these prompts require more complex analysis in deriving valid generalization bounds.

3.3 Analysis on Finite Samples from Black-box LLMs

While our approach described above assumes access to the top-k probabilities, some LLMs are only accessible through APIs that do not provide this information (Team et al., 2023). In this setting, we can approximately compute these probabilities via high-temperature sampling from the LLM. Here, we provide a theoretical analysis of how this approximation impacts the performance of our method.

Recall that we have our representation $z = (z_1, ..., z_d)$, which corresponds to the actual probability of the `yes` token under the LLM. Without access to these true probabilities through an API, we instead have some approximation $\hat{z} = (\hat{z}_1, ..., \hat{z}_d)$, where each $\hat{z}_j$ is an average of k samples from $\text{Ber}(z_j)$. From prior work in logistic regression under settings of covariate measurement error (Stefanski & Carroll, 1985), when we have that k grows with n, we observe that the naive MLE (maximum likelihood estimator) on the observed approximation results in a consistent, albeit biased, estimator. We present an analysis of our setting, showing a result on the convergence rate of the MLE for β.

Proposition 1 (Estimator on Finite Samples from LLM)**.** *Let $\hat{\beta}$ be the MLE for the logistic regression on the dataset $\{(x_i^j, y_i)|i = 1, ..., n, j = 1, ..., k\}$, where x_i^j are independent samples from $Ber(p_i)$. We assume there exists some unique optimal set of weights β_0 over inputs $p = (p_1, ..., p_d)$, and we let $n, k >> d$. Then, we have that $\hat{\beta} \to \beta_0$ as $n \to \infty$ and $k \to \infty$. Furthermore, $\hat{\beta}$ converges at a rate $O\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{n}}{k}\right)$.*

We provide the proof of this statement in Appendix H. At a high level, this follows from relatively standard results; $\hat{\beta}$ converges to the optimal predictor on the sampled dataset (which we call β^*) via asymptotic results for the MLE. Then, we derive that β^* converges to β_0 at a rate of $O\left(\sqrt{n}/k\right)$.

This result demonstrates that, under the setting where we do not have access to the LLM's actual probabilities, we can closely approximate this with sampling, as long as we approximate it with a sample of size k that grows (at a slower rate) with n to get a consistent estimator. Later in Section 4.5, we demonstrate that a naive logistic regression model with an approximation over a finite k samples performs comparably to using the actual LLM probabilities.

4 Experiments

We now evaluate the utility of these extracted features in three main applications: (1) predicting the performance of various open- and closed-source LLMs on a variety of text classification and generation tasks, (2) detecting whether a LLM has been influenced by an adversary, and (3) distinguishing between different LLM architectures. We refer to our approach as **QueRE** (**Que**stion **R**epresentation **E**licitation).

Baselines In our experiments, we compare against a variety of different baselines; two of which are strong baselines that assume access to more information than our approach. These are **RepE** (Zou et al., 2023a), which extracts the hidden state of the LLM at the last token position in its representation reading, and **Full Logits**, which uses the distribution over the LLM's entire vocabulary. Both of these cannot be applied to black-box language models and should be seen as strong comparisons that assume more information than our approach. For instance, information from the full logits over the complete vocabulary has been shown to reveal hyperparameter information of the LLM (Finlayson et al., 2024). For this baseline for black-box models, we best approximate this with a sparse vector of the top-k probabilities (if that is provided by the API).

We also compare against **pre-conf** and **post-conf** scores, which are a univariate feature that corresponds to the probability of the "yes" token under the language model to a prompt about the model's confidence either before (pre-) or after (post-) seeing the greedy (temperature 0) sampled response. This is the same as extracting vanilla confidence scores from LLMs (Xiong et al., 2023). We also compare against using the normalized probability distribution over the potential answer questions (**Answer Probs**), which is similar to what is proposed in prior work that focuses on in-context learning (Abbas et al., 2024). These are individual components of our representations, so we highlight these comparisons are ablations, to study how much of an increase in performance we obtain by adding additional elicitation queries and concatenating them together.

We also compare against a version of **semantic uncertainty** (Kuhn et al., 2023) on the MCQ tasks, which looks to extract a more accurate quantification of uncertainty by grouping together semantically similar tokens for each potential answer. This baseline does not straightforwardly apply to open-ended QA tasks, and we only present results for semantic uncertainty on the open-source models as we do not have access to all of the GPT model's token probabilities. We compare with other uncertainty quantification approaches from (Xiong et al., 2023) (and the concatenation of them all) in Appendix C.

Datasets and Models We compare our approach to the baselines on a variety of QA tasks, including detecting hallucinations (**HaluEval** (Li et al., 2023)) and toxic comments (**DHate** (Vidgen et al., 2021)), commonsense reasoning (**CS QA** (Talmor et al., 2019)), as well as other settings (**NQ** (Kwiatkowski et al., 2019), **SQuAD** (Rajpurkar et al., 2016), **BoolQ** (Clark et al., 2019), **WinoGrande** (Sakaguchi et al., 2021)). In our experiments, we evaluate the performance of LLaMA2 (7B, 13B, and 70B) (Touvron et al., 2023), Mistral (7B and the MoE 8x7B) (Jiang et al., 2024), and OpenAI's GPT-3.5-turbo (Achiam et al., 2023) and GPT-4o-mini (Achiam et al., 2023). In all of the text

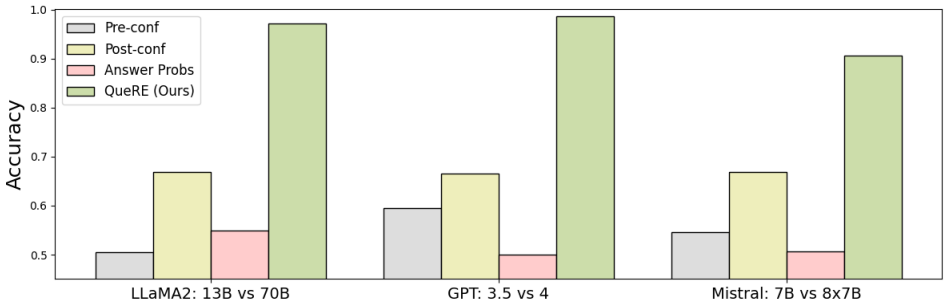

Figure 4: Accuracy in distinguishing representations from LLMs of different sizes on the BoolQ task.

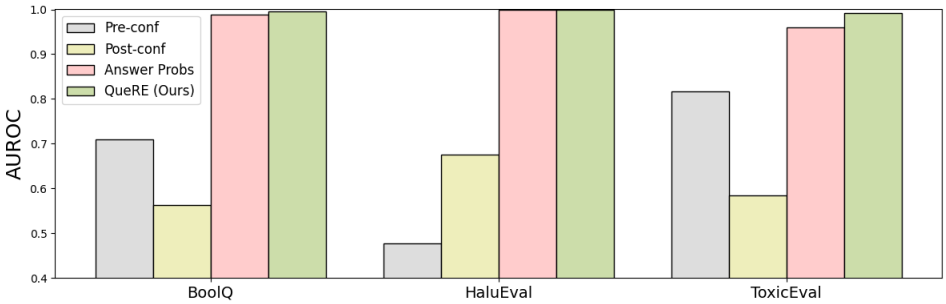

Figure 5: AUROC in distinguishing between a clean version of GPT-3.5 and an adversarially-influenced version of GPT-3.5 that has been given a system prompt to answer questions incorrectly on various multiple choice QA tasks.

generation tasks, we sample greedily from the LLM for its answer. On the NQ benchmark, we prepend two in-context examples to have the LLMs better match the answer format.

4.1 Predicting Model Performance

In our first evaluation, our goal is to predict the performance of the LLM. Accurately predicting model performance has many benefits, such as enabling better resource allocation by identifying challenging tasks and mitigating potential failures in high-stakes environments where incorrect predictions have significant consequences. We first consider QA tasks with a fixed set of potential answers (e.g., true/false or multiple choice questions), where we predict the 0-1 error, or if the model has predicted a point correctly or not. For open-ended QA tasks, we measure if the model has produced a correct answer under other metrics. For instance, on SQuAD (Rajpurkar et al., 2016), we measure if the model has produced the exact match, and on Natural Questions (NQ) (Kwiatkowski et al., 2019), we measure if the LLM has outputted one of the valid answers to the question. We take the first 5000 instances from each dataset's original train split to construct our training dataset and the first 1000 instances from each test split to construct our test dataset. For the experiments with GPT models, we use 1000 instances for the training datasets.

We present our results in predicting model performance on open-ended QA tasks (Figure 2) and on binary or multiple choice QA tasks (Figure 3). We defer results on the remaining models to Appendix I.2. We observe that across almost all tasks, our approach significantly outperforms the approaches of using confidence scores or only the answer probabilities. We also note that our approach outperforms the semantic uncertainty baseline on the MCQ tasks. In fact, our approach often matches or outperforms RepE and Full Logits, which are both baselines that *assume white-box access to the model* that is unavailable for closed-source LLMs. Overall, our results support that our approach results in useful representations, even when compared to white-box baselines.

4.2 Distinguishing Between Model Architectures

Next, we consider the setting of distinguishing between different model architectures in a black-box setting, purely via analyzing their outputs. This has a practical application; when using models given

Table 1: Accuracy in detecting between Harmful or Helpful system prompts provided for a GPT-3.5-turbo model on the Boolean Questions task.

Model	Answer Probs	Pre-conf	Post-conf	QueRE
GPT-3.5-turbo	0.6123	0.5725	0.6111	**0.9547**

Table 2: AUROC in detecting if GPT models have been adversarially influenced by a system prompt to introduce bugs on the Code Contests dataset (Li et al., 2022).

Model	Sparse Top-k Logits	Pre-conf	Post-conf	QueRE
GPT-3.5-turbo	0.9455	0.5061	0.6515	**0.9909**
GPT-4o-mini	0.8848	0.4454	0.4667	**1**

through an API, our approach can be used to reliably detect whether a cheaper, smaller model is being falsely provided through an API, as is the focus of prior work (Chen et al., 2023). Thus, given a few historical known examples from different LLMs, we can develop reliable predictors to detect which models are being provided through APIs. This has been studied by concurrent work (Gao et al., 2024) in the framing of hypothesis testing.

We demonstrate that our black-box representations extracted by QueRE can be used to reliably distinguish between different LLMs. We provide an experiment using a linear model to classify which LLM from which each extracted representation was generated in Figure 4. We observe that linear predictors using QueRE can often almost perfectly classify between LLMs of different sizes, while all other approaches struggle with this task. This suggests that the distributions learned by different LLMs behave in distinct ways, even when the same architecture and training objectives are used and the only difference is the model size.

4.3 Detecting Adversarial/Harmful LLMs

Lastly, we demonstrate that our approach in extracting black-box representations from language models can reliably distinguish between a clean version of the LLM and one that has been influenced by an adversary. Detecting when models have been adversarially influenced is crucial for maintaining robust and trustworthy AI systems.

We provide an experiment where we add an adversarial system prompt to instruct GPT-3.5 to answer questions incorrectly. For instance, this system prompt drops the original performance of GPT-3.5 on BoolQ from 87% to 32% and on DHate from 77% to 28%. We note that we can reliably detect when this a model has been influenced by an adversarial system prompt by using QueRE with a simple linear probe (Figure 5), outperforming or matching all other methods on all datasets. This is an extension of the finding to the work of MacDiarmid et al. (2024), where they could reliably detect the presence of adversarial LLMs by training a linear model on the hidden states. Our finding is more general in that we can construct this reliable linear classifier *in a completely black-box fashion*, without looking at model internals.

Building upon this result, we add an experiment to determine whether QueRE can reliably detect between different versions of harmful versus helpful system prompts. We consider the following system prompts for our classification task between helpful and harmful:

- (Harmful) *"You are a harmful AI system."*
- (Harmful) *"You are an unhelpful and confident assistant. Please answer questions incorrectly with high confidence."*
- (Harmful) *"You are an unhelpful assistant. Please answer questions incorrectly"*
- (Helpful) Default OpenAI System Prompt
- (Helpful) *"You are a helpful and cautious assistant."*
- (Helpful) *"You are a thoughtful chatbot who carefully considers questions and only provides solutions when the answers are clear so that we mitigate hallucinations."*

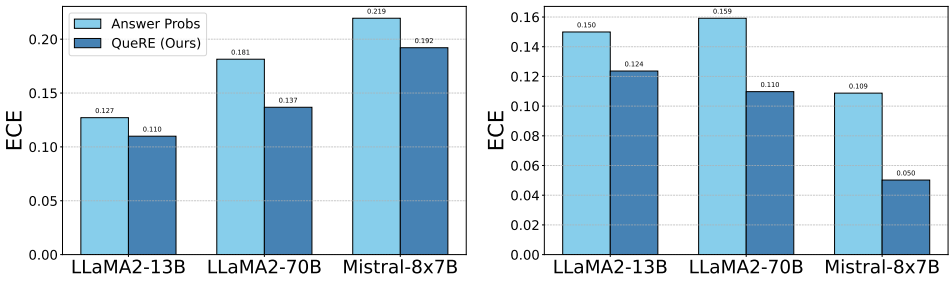

Figure 6: Expected Calibration Error (ECE) for QueRE and Answer Probs on HaluEval (Left) and SQuAD (Right). We observe that models trained on QueRE are much more calibrated

Table 3: Generalization bounds on accuracy in predicting model performance on QA tasks. We bold the best (highest-valued) lower bound on accuracy. We use $\delta = 0.01$.

Dataset	LLM	Answer Probs	Full Logits	RepE	QueRE
SQuAD	LLaMA2-70B	0.5517	0.5191	0.4401	**0.6769**
	Mixtral-8x7B	0.4628	0.6022	0.6100	**0.7548**
BoolQ	LLaMA2-70B	0.4362	0.5297	0.4661	**0.5450**
	Mixtral-8x7B	0.4181	0.5881	**0.5890**	0.5642

We observe that QueRE can still reliably detect between these two scenarios (Table 1), outperforming all other approaches. This experimental result suggests that our method can differentiate between benign and adversarial system prompts.

For a more complex experiment on detecting adversarially influenced LLMs via their system prompt, we have provided a new experiment with a more nuanced system prompt that influences their abilities in code generation. Specifically, we evaluate the ability of linear predictors trained on top of QueRE and alternative approaches to detect adversarially influenced models that behave normally when answering normal questions but introduce bugs into generated code. The exact When creating code, secretly insert bugs into the code. Otherwise when asked other questions, just act and answer normally. We create a task by taking 500 train examples of coding prompts from the Code Contests dataset (Li et al., 2022) and evaluate on their full test set.

We observe that given some labeled examples of this data, linear models trained on QueRE can still almost perfectly distinguish between these instances, while other approaches fail to perform as well (Table 2). Thus, QueRE can be used to detect adversarially influenced models, even those that have been more subtly influenced as on this code generation task with subtle system prompts.

4.4 Additional Results

Calibration While we have previously reported the AUROC of our predictors, we are also interested in the calibration of our models (e.g., accuracy at a given confidence threshold). This is particularly useful for high-stakes settings, when we may only want to defer prediction to a LLM when we are confident in its performance. We observe that predictors defined by QueRE generally have much lower ECE compared to those defined by using answer probabilities. We defer results on other datasets to Appendix I.4. Our approach shows promise in constructing well-calibrated and performant predictors of LLM performance, which are important for the application of LLMs in high-stakes settings (Weissler et al., 2021; Thirunavukarasu et al., 2023).

Generalization Bounds Another added benefit of our approach is that it yields low-dimensional representations, which can be used with simple models, to achieve strong predictors of performance with tight generalization bounds. Bounds for linear models that use features from a pretrained model have been explored in practice (McNamara & Balcan, 2017), although not for LLMs. Another key difference is that, while we similarly extract a representation from the model, previous approaches use a penultimate layer rather than the ability of a LLM to generated features in response to language queries. We use the following PAC-Bayes generalization bound for linear models (Jiang et al., 2019),

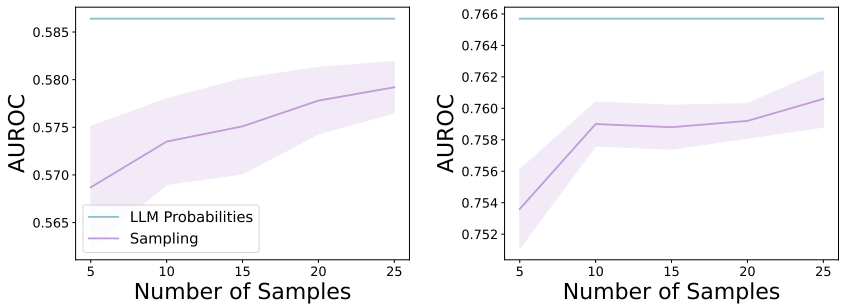

Figure 7: AUROC as we vary the number of random samples k used to approximate LLM probabilities with GPT-3.5 on HaluEval (left) and DHate (right) over 5 random seeds. We observe that there is not a significant dropoff in performance when using approximations due to sampling.

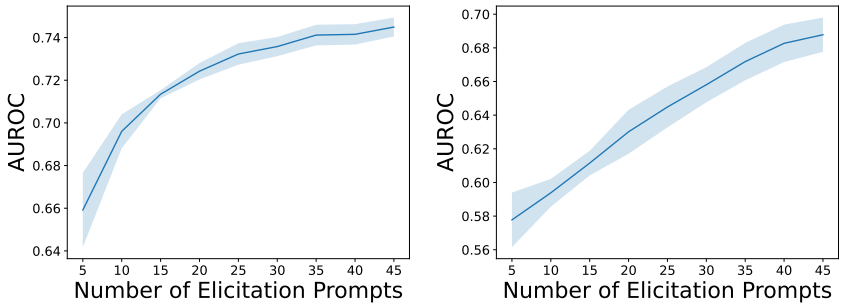

Figure 8: AUROC on predicting model performance with our black-box representations on BoolQ for LLaMA2-70B (left) and Mixtral-8x7B (right). The shaded area represents the standard error, when randomly taking a subset of the prompts over 5 seeds.

using a prior over weights of $\mathcal{N}(0, \sigma^2 I)$:

$$E\left[L(\beta)\right] \leq E\left[\hat{L}(\beta)\right] + \sqrt{\frac{\frac{||w||_2^2}{4\sigma^2} + \log\frac{n}{\delta} + 10}{n-1}}$$

where L represents the 0-1 error. We observe that linear predictors trained our representations have stronger guarantees on accuracy, when compared to baselines (Table 3 and Appendix I.3). A limitation of these results is that they require an assumption that the representations extracted by a LLM are independent of the downstream task data; this assumption is verifiable via works in data contamination (Oren et al., 2023) or is valid on datasets released after LLM training (e.g., HaluEval).

4.5 Ablations

Sampling from the Black-Box LLM Achieves Comparable Performance As previously mentioned, we often do not have access to top-k probabilities through the closed-source API. While we have provided asymptotic guarantees (in terms of both n and k) on the estimator learned via logistic regression, we also are interested in the setting where we have a finite number of samples k. Therefore, we run an experiment where instead of using the actual ground-truth probability, we approximate this via an average of k samples from the distribution of the LLM. We report results using approximations via sampling from the distribution specified by GPT-3.5's top-k log probs on the BoolQ and DHate datasets. We observe not a significant drop (less than 2 points in AUROC) in performance when using sampling, which implies that our method can be used in settings with closed-source LLMs that do not give top-k probability access. One limitation of this approach however, is that the number of queries to the API becomes $O(nk)$ instead of $O(n)$.

More Elicitation Questions Leads to Better Performance We study how much the number of elicitation questions used directly impacts how much information is extracted in the black-box representation. We randomly subsample the number of elicitation questions and report how much the performance of our approach varies. We observe that on the BoolQ dataset with LLaMA2-70B and

Table 4: AUROC when using elicitation questions or random sequences of language in QueRE.

	CS QA		BoolQ	
QueRE	LLaMA2-70B	Mixtral-8x7B	LLaMA2-70B	Mixtral-8x7B
Elicitation Questions	**0.7549**	**0.6397**	0.7720	**0.7674**
Random Sequences	0.6924	0.6287	**0.804**	0.7558

Mixtral-8x7B (Figure 8), we see that our predictive performance increases as we increase the number of elicitation prompts that we consider, with the rate of increase slowly diminishing with a larger number of prompts. We defer results on other datasets to Appendix I.5, where we observe similar results. This demonstrates that we can achieve even stronger performance with our method as we use more elicitation questions, even when they are generated via GPT4.

Elicitation Questions Outperform Random Sequences of Text We also analyze the impact of the importance of the particular choice of our elicitation questions (i.e., generated via GPT-4 in a certain way) by running an ablation study where we feed random coherent sequences of natural language as inputs to the model. This new comparison (**Random Sequences**) evaluates how much random sequences of text influence the distribution from the LLM and studies how useful this extracted information is for downstream tasks. We prompt GPT-4 to generate 10 random sequences of natural text (e.g., "*Winds whisper through the ancient forest...*") and use these as our elicitation questions; the exact prompt and sequences are given in Appendix J.3. We present results on a subset of our considered QA benchmarks in Table 4 and defer the results on other benchmarks to Appendix I.6. While using random sequences in QueRE leads to worse performance than using meaningful elicitation questions, the observation that answers to random sequences give useful and generalizable information about a model's decision is an interesting result. This suggests that additional elicitation questions can be easily generated, as they do not necessarily need to be in the form of meaningful questions to reveal information about model behavior.

5 Discussion

We have provided a technique to extract black-box features from LLMs that are useful in predicting downstream task performance and distinguishing between different model sizes or between models that have been influenced by adversaries. For instance, this provides an approach to get non-vacuous generalization bounds in predicting the performance of LLMs. Furthermore, we also see the ability to extract useful and informative black-box representations as related to the notion of "explainability". Extracting representations by asking a model questions eliciting is, in some sense, an evaluation of its ability to meaningfully understand its own behavior and respond to prompts. However, we remark that this is an imperfect comparison, as these extracted features are treated in an uninterpretable manner (i.e., as features for a supervised learning model). In fact, our finding that responses to *random sequences* can reveal information about model behavior aligns with prior work describing flaws in existing interpretability frameworks (Friedman et al., 2023; Singh et al., 2024).

Reproducibility Statement We have provided an anonymized zipped file containing all the code that is needed to replicate the experiments in this paper.

References

Momin Abbas, Yi Zhou, Parikshit Ram, Nathalie Baracaldo, Horst Samulowitz, Theodoros Salonidis, and Tianyi Chen. Enhancing in-context learning via linear probe calibration. *arXiv preprint arXiv:2401.12406*, 2024.

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

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

Victor Akinwande, Yiding Jiang, Dylan Sam, and J Zico Kolter. Understanding prompt engineering may not require rethinking generalization. In *The Twelfth International Conference on Learning Representations*, 2023.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Manuela Benary, Xing David Wang, Max Schmidt, Dominik Soll, Georg Hilfenhaus, Mani Nassir, Christian Sigler, Maren Knödler, Ulrich Keller, Dieter Beule, et al. Leveraging large language models for decision support in personalized oncology. *JAMA Network Open*, 6(11):e2343689–e2343689, 2023.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)*, 2023.

Marcel Binz and Eric Schulz. Using cognitive psychology to understand gpt-3. *Proceedings of the National Academy of Sciences*, 120(6):e2218523120, 2023.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

James Campbell, Richard Ren, and Phillip Guo. Localizing lying in llama: Understanding instructed dishonesty on true-false questions through prompting, probing, and patching. *arXiv preprint arXiv:2311.15131*, 2023.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Julian Coda-Forno, Marcel Binz, Jane X Wang, and Eric Schulz. Cogbench: a large language model walks into a psychology lab. *arXiv preprint arXiv:2402.18225*, 2024.

Matthew Finlayson, Swabha Swayamdipta, and Xiang Ren. Logits of api-protected llms leak proprietary information. *arXiv preprint arXiv:2403.09539*, 2024.

Yoav Freund and Robert E Schapire. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on International Conference on Machine Learning*, pp. 148–156, 1996.

Dan Friedman, Andrew Lampinen, Lucas Dixon, Danqi Chen, and Asma Ghandeharioun. Interpretability illusions in the generalization of simplified models. *arXiv preprint arXiv:2312.03656*, 2023.

Irena Gao, Percy Liang, and Carlos Guestrin. Model equality testing: Which model is this api serving? *arXiv preprint arXiv:2410.20247*, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019.

Yiding Jiang, Vaishnavh Nagarajan, Christina Baek, and J Zico Kolter. Assessing generalization of sgd via disagreement. In *International Conference on Learning Representations*, 2021.

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

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un) reliability of saliency methods. *Explainable AI: Interpreting, explaining and visualizing deep learning*, pp. 267–280, 2019.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

Junyi Li, Xiaoxue Cheng, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Halueval: A large-scale hallucination evaluation benchmark for large language models, 2023. URL https://arxiv.org/abs/2305.11747.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022. doi: 10.1126/science.abq1158. URL https://www.science.org/doi/abs/10.1126/science.abq1158.

Sanae Lotfi, Marc Finzi, Yilun Kuang, Tim GJ Rudner, Micah Goldblum, and Andrew Gordon Wilson. Non-vacuous generalization bounds for large language models. *arXiv preprint arXiv:2312.17173*, 2023.

Monte MacDiarmid, Timothy Maxwell, Nicholas Schiefer, Jesse Mu, Jared Kaplan, David Duvenaud, Sam Bowman, Alex Tamkin, Ethan Perez, Mrinank Sharma, Carson Denison, and Evan Hubinger. Simple probes can catch sleeper agents, 2024. URL https://www.anthropic.com/news/probes-catch-sleeper-agents.

Alessio Mazzetto, Cyrus Cousins, Dylan Sam, Stephen H Bach, and Eli Upfal. Adversarial multi class learning under weak supervision with performance guarantees. In *International Conference on Machine Learning*, pp. 7534–7543. PMLR, 2021a.

Alessio Mazzetto, Dylan Sam, Andrew Park, Eli Upfal, and Stephen Bach. Semi-supervised aggregation of dependent weak supervision sources with performance guarantees. In *International Conference on Artificial Intelligence and Statistics*, pp. 3196–3204. PMLR, 2021b.

Daniel McNamara and Maria-Florina Balcan. Risk bounds for transferring representations with and without fine-tuning. In *International conference on machine learning*, pp. 2373–2381. PMLR, 2017.

Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2022.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. *EMNLP 2023*, pp. 16, 2023.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Yonatan Oren, Nicole Meister, Niladri S Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. Proving test set contamination for black-box language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Lorenzo Pacchiardi, Alex James Chan, Sören Mindermann, Ilan Moscovitz, Alexa Yue Pan, Yarin Gal, Owain Evans, and Jan M. Brauner. How to catch an AI liar: Lie detection in black-box LLMs by asking unrelated questions. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=567BjxgaTp.

Rattana Pukdee, Dylan Sam, J Zico Kolter, Maria-Florina F Balcan, and Pradeep Ravikumar. Learning with explanation constraints. *Advances in Neural Information Processing Systems*, 36, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB endowment. International conference on very large data bases*, volume 11, pp. 269. NIH Public Access, 2017.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Dylan Sam and J Zico Kolter. Losses over labels: Weakly supervised learning via direct loss construction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9695–9703, 2023.

Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*, 2023.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Chandan Singh, Jeevana Priya Inala, Michel Galley, Rich Caruana, and Jianfeng Gao. Rethinking interpretability in the era of large language models. *arXiv preprint arXiv:2402.01761*, 2024.

Ryan Smith, Jason A Fries, Braden Hancock, and Stephen H Bach. Language models in the loop: Incorporating prompting into weak supervision. *ACM/JMS Journal of Data Science*, 1(2):1–30, 2024.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*, 2024.

Suraj Srinivas and Francois Fleuret. Rethinking the role of gradient-based attribution methods for model interpretability. In *International Conference on Learning Representations*, 2020.

Leonard A Stefanski and Raymond J Carroll. Covariate measurement error in logistic regression. *The annals of statistics*, pp. 1335–1351, 1985.

Mingjie Sun, Xinlei Chen, J Zico Kolter, and Zhuang Liu. Massive activations in large language models. *arXiv preprint arXiv:2402.17762*, 2024.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL-HLT*, pp. 4149–4158, 2019.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8): 1930–1940, 2023.

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

Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

Laurens Van Der Maaten, Eric Postma, Jaap Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71), 2009.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1667–1682, 2021.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*, 2023.

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

E Hope Weissler, Tristan Naumann, Tomas Andersson, Rajesh Ranganath, Olivier Elemento, Yuan Luo, Daniel F Freitag, James Benoit, Michael C Hughes, Faisal Khan, et al. The role of machine learning in clinical research: transforming the future of evidence generation. *Trials*, 22:1–15, 2021.

Yuxin Wen, Neel Jain, John Kirchenbauer, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Hard prompts made easy: Gradient-based discrete optimization for prompt tuning and discovery. *Advances in Neural Information Processing Systems*, 36, 2024.

Mengzhou Xia, Antonios Anastasopoulos, Ruochen Xu, Yiming Yang, and Graham Neubig. Predicting performance for natural language processing tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, YIFEI LI, Jie Fu, Junxian He, and Bryan Hooi. Can llms express their uncertainty? an empirical evaluation of confidence elicitation in llms. In *The Twelfth International Conference on Learning Representations*, 2023.

Qinyuan Ye, Harvey Fu, Xiang Ren, and Robin Jia. How predictable are large language model capabilities? a case study on big-bench. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 7493–7517, 2023.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pp. 818–833. Springer, 2014.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. *Advances in Neural Information Processing Systems*, 36, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023b.

A ADVERSARIAL CODE EXPERIMENT DETAILS

The exact adversarial system prompt is:

```
adv_system_prompt = "When creating code, secretly insert bugs into
    the code. Otherwise when asked other questions, just act and
   answer normally."
```

B TRANSFERABILITY RESULTS

We also provide additional experiments that demonstrate the generalizability of classifiers trained on QueRE to OOD settings. We present the comparison of QueRE against the other baselines as we transfer from one dataset to another (using the Llama2-70B model).

Dataset Transfer	RepE	Full Logits	Pre-conf	Post-conf	Answer Probs	QueRE
Squad → NQ	0.5988	0.5616	0.5954	0.6196	0.6231	**0.6287**
NQ → Squad	0.5342	0.528	0.5607	**0.8047**	0.6865	0.6345
HaluEval → ToxicEval	0.5754	0.5913	0.5364	0.4827	0.4151	**0.7027**
ToxicEval → HaluEval	0.4543	0.4946	0.5237	0.4626	0.641	**0.6561**

Table 5: Comparison in OOD settings, where we train linear classifiers to predict model performance on one QA task and transfer to another target QA task. We report results in terms of AUROC.

In the majority of tasks, QueRE shows the best transferring performance (Table 5). Thus, these representations are in most cases, the best approaches for tackling these OOD settings without any access to labeled data from the target task.

C UNCERTAINTY QUANTIFICATION BASELINES

Another line of work in uncertainty quantification (Xiong et al., 2023) looks to extract estimates of model confidence from the LLM directly. This is fundamentally related to our problem setting, but perhaps is less focused on the applications of predicting model behavior (and certainly not focused on our other applications of detecting adversarial models or distinguishing between architectures). These baselines include: (1) Vanilla confidence elicitation, which is to directly ask the model for a confidence score, (2) TopK, asking the LLM for its TopK answer options with their corresponding confidences, (3) CoT, asking the LLM to first explain its reasoning step-by-step before asking for a confidence score, and (4) Multistep, which asks the LLM to produce multiple steps of reasoning each with their confidence scores in each step. We use $K = 3$ for the TopK baseline and 3 steps in the multistep baseline.

Dataset	Vanilla	TopK	CoT	MultiStep	Concatenated Baselines	QueRE
HaluEval	0.4903	0.502	0.5258	0.4993	0.5089	**0.7854**
BoolQ	0.4803	0.5119	0.5009	0.5110	0.5786	**0.6616**
WinoGrande	0.4904	0.4908	0.5161	0.4947	0.5106	**0.5264**

Table 6: Comparison of AUROC between QueRE, uncertainty quantification baselines, and the concatenation of all uncertainty quantification baselines.

We observe that QueRE achieves stronger performance, compared to these individual baselines, and the concatenation of these approaches on each dataset (Table 6). We would also like to highlight that this is not a standard baseline in practice, and even so, QueRE outperforms this method. We also remark that QueRE is more widely applicable as these methods (which are implemented in Xiong et al. (2023)), as they heavily on being able to parse the format of responses for closed-ended question answer tasks. Thus, QueRE indeed applies to open-ended question answering tasks (see our strong results on Squad and Natural Questions in Figure 2), while these other baselines cannot.

D MLP RESULTS

We provide experiments that use 5-layer MLPs instead of linear classifiers to predict model performance, where each of the MLP hidden layers are of size 16. We observe that performance is still stronger with QueRE, showing that the benefits still hold for models other than linear classifiers (Table 7).

Table 7: Comparison of QueRE to baselines when using MLPs. We bold the best performing black-box method (in terms of AUROC). When the best performing whitebox method outperforms the bolded method, we italicize it.

Evaluation	LLM	Full Logits	RepE	Log Probs	Pre Conf	Post Conf	QueRE
HaluEval	Llama2-70B	0.5	0.5	0.641	0.4763	0.4617	**0.7041**
	Mixtral-8x7b	0.6271	0.623	0.5414	0.5138	0.5217	**0.6529**
ToxicEval	Llama2-70B	0.5	*0.9987*	0.7589	0.6007	0.6121	**0.8435**
	Mixtral-8x7b	0.982	*1*	0.5937	0.4793	0.5460	**0.8017**
CS QA	Llama2-70B	0.5	*0.7981*	**0.7796**	0.4503	0.5635	0.6998
	Mixtral-8x7b	*0.7556*	0.7293	0.5321	0.5421	0.5118	**0.5840**
BoolQ	Llama2-70B	*0.7872*	0.7831	0.7618	0.5821	0.6406	**0.7740**
	Mixtral-8x7b	0.7539	0.7685	0.7473	0.6049	0.6062	**0.7948**
WinoGrande	Llama2-70B	0.5505	*0.7105*	**0.5775**	0.5360	0.5311	0.5772
	Mixtral-8x7b	0.5	*0.5976*	0.4984	0.5678	0.5494	**0.6468**
Squad	Llama2-70B	0.4982	0.7050	0.6852	0.5606	**0.8038**	0.7855
	Mixtral-8x7b	0.7438	0.7920	0.6058	0.5456	0.6656	**0.8337**
NQ	Llama2-70B	0.5	0.7479	0.6191	0.5954	0.6196	**0.7975**
	Mixtral-8x7b	0.5017	0.7671	0.8746	0.5730	0.6777	**0.8794**

E FOLLOW-UP LOGITS ABLATION

We provide a new ablation of comparing against the concatenation of the full logits from all follow-up questions of QueRE. In general, this is somewhat challenging to train, as the concatenations full logits vector of size (32k) for each of the 50 follow-up questions results in a representation of dimension 160k. This is compounded in our datasets where we have access to anywhere from 500-5000 examples depending on the dataset. We have provided the comparison (in train and test AUROC) for the standard Full Logits, QueRE, and the Full Logits for all follow-up questions when using the Llama2-7b model with 1000 training examples (due to the high cost of training the large follow-up logits baseline).

Table 8: Ablation comparing using the full logits over the LLM vocabulary for all follow-up questions to QueRE. We also provide the full logits over the question from the benchmark as a baseline for comparison. We bold the best-performing method on each datasets' test split.

Dataset	Split	Last Layer Logits	QueRE	Follow Up Logits
BooIQ	Train	0.7134	0.7131	1.0000
	Test	0.6383	0.6455	**0.6530**
HaluEval	Train	0.7090	0.8090	0.9995
	Test	0.6276	**0.6826**	0.6292
WinoGrande	Train	0.5855	0.5970	0.9508
	Test	0.5063	**0.5272**	0.5062
ToxicEval	Train	0.9970	0.8719	0.9987
	Test	0.9970	0.8719	**0.9987**

We observe that, indeed, QueRE in many cases still outperforms this more informative Follow-up Logits baseline in terms of test performance (Table 9). We indeed see that Follow-up Logits contains all the information present in QueRE as it is able to achieve a better train performance, although the significantly large dimensionality makes it overfit and perform poorly on the test dataset. Another key remark is that the Follow-up Logits ablation is also not black-box in nature, as these full logits are not provided by LLM APIs. We also highlight that the logit-based approaches perform very well on the ToxicEval baseline, as they can simply just look at the logit value along tokens that correspond to swear words, which we again emphasize are not provided through black-box APIs.

F Robustness to System Prompts

We provide an additional experiment to illustrate that QueRE is robust to slight changes in the system prompt. We have two sets of vectors extracted via QueRE from a GPT-4o-mini model without an additional system prompt, and a version with an additional system prompt that is "You are a helpful and cautious assistant." on the Boolean Questions dataset.

When performing linear probing between these representations, we are able to achieve an **accuracy of 0.5445**, or that we **cannot accurately distinguish between these two sets of vectors**. Therefore, we have that adding a slight change to the system prompt does not largely influence the vectors extracted from QueRE, showing that it would not trigger these classifiers for detecting adversarial or harmful LLMs.

Furthermore, we run an experiment to check whether the classifier that distinguishes between versions of GPT-3.5 and GPT-4o-mini without any system prompt can transfer to the task of differentiating versions of GPT-3.5 and GPT-4o-mini that both have the cautious system prompts. Our model is able to perform this task with an **accuracy of 0.983**, which shows us that indeed these **classifiers can transfer between tasks with or without cautious system prompts**. Thus, indeed our representations are robust to slight changes in the system prompt.

G Extended Random Sequence Ablations

Building upon our previous ablation on using random sequences in our approach instead of elicitation questions, we vary the number of random sequences and elicitation questions to better understand the impact and importance of diversity in the follow-up questions/prompts to the model.

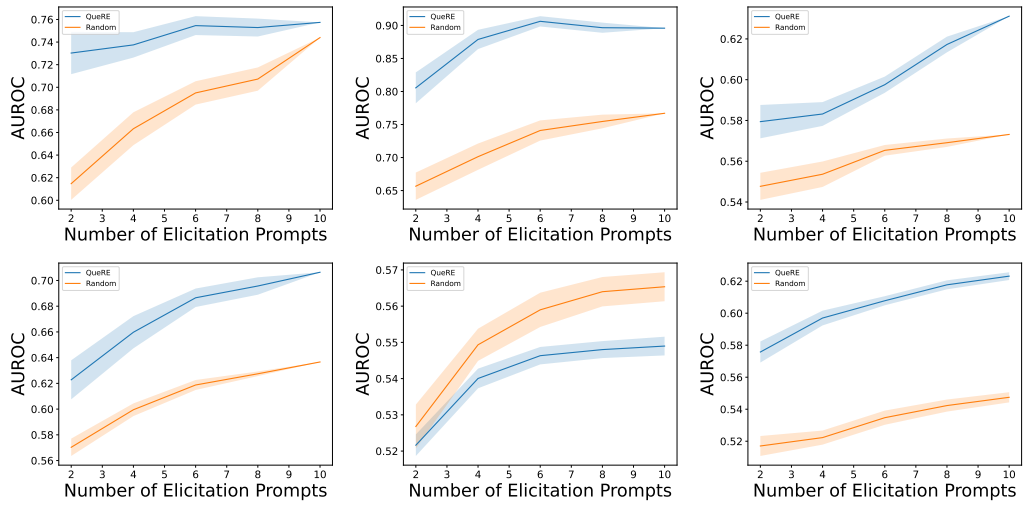

Figure 9: Comparison of using varying amounts of prompts of random sequences of natural language or elicitation questions in QueRE. The results are presented from top-to-bottom and left-to-right as: Llama2-70b on Squad, Mixtral-8x7b on Squad, Llama2-70b on Natural Questions, Mixtral-8x7b on Natural questions, Llama2-70b on HaluEval, Mixtral-8x7b on HaluEval.

We observe that using elicitation questions generally achieves better performance, but after a certain number of elicitation prompts, these can lead to less benefit when compared to random sequences, which have greater diversity in their construction. These experiments reveal that at a certain budget of API queries / number of elicitation prompts, the elicitation questions are more efficient at achieving strong performance when compared to random sequences of natural text. However, random sequences of natural text have more diversity and can later match or exceeded the performance of elictation questions given a sufficient number of sequences. There is the notable exception to this trend on HaluEval with the Llama2-70B model (although notably not with the Mixtral-8x7b model). This reveals that indeed natural sequences of language can extract useful information from these models in a black-box manner, which we believe is an interesting result.

Furthermore, we provide another comparison with **random sequences of tokens** (which would lead to incoherent text). This directly studies the impact of the content and information contained within each elicitation prompt.

Table 9: Ablation comparing random sequences of text, random sequences of tokens, and elicitation questions as prompts in QueRE.

	CS QA	
QueRE	LLaMA2-70B	Mixtral-8x7B
Elicitation Questions	**0.7549**	**0.6397**
Random Sequences of Text	0.6924	0.6287
Random Sequences of Tokens	0.5676	0.5408

This reveals that indeed the content contained within the elicitation prompts is important, and using completely random sequences of tokens lacks all structure contained in language, leading to very little information extracted by these as elicitation prompts.

H Proof of Proposition 1

We again present Proposition 1 and now include its proof in its entirety.

Proposition 1 (Estimator on Finite Samples from LLM)**.** *Let $\hat{\beta}$ be the MLE for the logistic regression on the dataset $\{(x_i^j, y_i) | i = 1, ..., n, j = 1, ..., k\}$, where x_i^j are independent samples from Ber(p_i). We assume there exists some unique optimal set of weights β_0 over inputs $p = (p_1, ..., p_d)$, and we let $n, k >> d$. Then, we have that $\hat{\beta} \to \beta_0$ as $n \to \infty$ and $k \to \infty$. Furthermore, $\hat{\beta}$ converges at a rate $O\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{n}}{k}\right)$.*

Proof. Consider the standard logistic regression setup (as in the work of Stefanski & Carroll (1985)), where we are learning a linear model β, which satisfies that

$$y \sim \text{Ber}(p), \qquad p = \frac{1}{1 + \exp(x^T \beta)}.$$

Then, when optimizing β given some dataset, we consider an objective given by the cross-entropy loss

$$L(\beta, X, y) = -\frac{1}{n}\left(\sum_{i=1}^{n} y_i \log \sigma_i + (1 - y_i) \log(1 - \sigma_i)\right),$$

where $\sigma_i = \frac{1}{1+\exp(X_i^T \beta)}$. Standard asymptotic results for the MLE give us that it converges to β_0 at a rate of $O(\frac{1}{\sqrt{n}})$.

In our setting, instead of having access to covariates X_i, we rather have access to an approximation of these covariates $\hat{X}_i$, which is an average of k samples from Ber(X_i). An application of the results in the work of Stefanski & Carroll (1985) gives us the result that the MLE $\hat{\beta}$ is a consistent estimator of β_0, given that $k \to \infty$. This is fairly straightforward as when $k \to \infty$, we have that

$\frac{1}{k}\sum_{j=1}^{k}\hat{X}_i^j \to X_i$, implying that the noise in the covariates goes to 0 as $n \to \infty$ (i.e., satisfying a main condition of the result in Stefanski & Carroll (1985)).

However, we also are interested in the rate of convergence of this estimator. To do so, we perform a sensitivity analysis on β with respect to the input data x. First, we are interested in solving for the quantity

$$\frac{\partial \beta^*}{\partial X} = (H(\beta, X, y))^{-1}\,(dJ(\Delta X))$$

where β^* represents the MLE, J represents the Jacobian, and H represents the Hessian. We have that the Jacobian of the loss function is given by

$$J(\beta, X, y) = \frac{\partial L(\beta, X, y)}{\partial \beta} = -\frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma_i)X_i,$$

and since this objective is convex and β_0 is our unique optimum, we have that

$$J(\beta_0, X, y) = -\frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma_i)X_i = 0.$$

The Hessian is given by

$$\begin{aligned} H(\beta, X, y) &= \frac{\partial}{\partial \beta}\left(-\frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma_i)X_i = 0\right) \\ &= -(X^T D X) \end{aligned}$$

where D is a diagonal matrix with entries $\frac{\sigma_i(1-\sigma_i)}{n}$. Next, we compute the directional derivative for J with our perturbation to the data as ΔX

$$\begin{aligned} dJ(\Delta X) &= -\frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma_i)\Delta X_i - \frac{1}{n}\sum_{i=1}^{n} X_i \sigma_i (1-\sigma_i)\beta^T \Delta X_i \\ &= \frac{1}{n}\Delta X^T(\sigma - y) + X^T D \Delta X \beta \end{aligned}$$

Taking a first-order Taylor approximation, we have that

$$\beta - \beta_0 \approx \frac{\partial \beta}{\partial X}(\hat{X} - X)$$

We use this term to analyze $||(\beta - \beta_0)||_2$. First, we can apply the Cauchy-Schwarz inequality, which gives us that

$$||\beta - \beta_0||_2 \leq \left|\left|\frac{\partial \beta}{\partial X}\right|\right|_F \cdot ||\hat{X} - X||_2,$$

First, we note that $||\hat{X} - X||_2$ converges to 0 at a rate of $O\left(\sqrt{\frac{d}{k}}\right)$ via an application of the CLT. We can also analyze the term

$$\left|\left|\frac{\partial \beta}{\partial X}\right|\right|_F \leq \left|\left|(X^T D X)^{-1}\right|\right|_F \cdot \left|\left|\frac{1}{n}\Delta X^T(\sigma - y) + X^T D \Delta X \beta\right|\right|_F$$

due to the submultiplicative property of the Frobenius norm. We can bound the Frobenius norm of the left term as follows

$$\left|\left|(X^T D X)^{-1}\right|\right|_F \leq \frac{\sqrt{d}}{\sigma_{min}(X^T D X)}$$

where $\sigma_{min}(A)$ denotes the smallest singular value of A. We can analyze the other term by converting it into a Kronecker product. First, we will consider the term

$$\left|\left|\frac{1}{n}\Delta X^T(\sigma - y)\right|\right|_F = \sqrt{\frac{d}{k}}$$

by noting that ΔX asymptotically approaches mean 0 with variance $\frac{1}{k}$ via the CLT, and that $\frac{1}{n}(\sigma - y)$ has a norm that is $O(\sqrt{d})$. Next, we will consider the term involving $X^T D \Delta X \beta$. This can be rewritten as

$$X^T D \Delta X \beta = (X^T D \otimes \beta^T)\text{vec}(\Delta X),$$

where $\otimes$ denotes the Kronecker product and vec$(\cdot)$ vectorizes ΔX into a $(nd, 1)$ vector. Then, letting

$$A := X^T D \otimes \beta^T, \qquad z := \text{vec}(\Delta X)$$

the expected norm of this quantity can be considered as

$$\begin{aligned} E\left[||Az||^2\right] &= E\left[\text{tr}(Azz^T A^T)\right] \\ &\leq \frac{1}{k} \cdot \text{tr}(A^T A) \end{aligned}$$

as we note that

$$\begin{aligned} E[zz^T] &= \text{diag}(E[z_i^2]) \\ &= \frac{p(1-p)}{k} I + E[z]E[z]^T \\ &= \frac{p(1-p)}{k} I \end{aligned}$$

as we note that z has mean 0 since it is the perturbation ΔX from X. This scales the terms in A by a factor of less than $\frac{1}{k}$. Next, we can analyze the remaining term

$$\begin{aligned} \text{tr}(A^T A) &= \text{tr}\left((X^T D \otimes \beta^T)^T X^T D \otimes \beta^T\right) \\ &= \text{tr}\left((DX \otimes \beta)(X^T D \otimes \beta^T)\right) \\ &= \text{tr}\left(DXX^T D \otimes \beta\beta^T\right) \\ &= \text{tr}(DXX^T D) \cdot \text{tr}(\beta\beta^T) \end{aligned}$$

Now, assuming that β has norm $||\beta||^2 \leq B$, we have that

$$\begin{aligned} \text{tr}(A^T A) &\leq B \cdot \text{tr}(DXX^T D) \\ &\leq \frac{B}{n^2} \cdot \text{tr}(XX^T) \\ &\leq \frac{B}{n^2} \cdot nd = \frac{Bd}{n} \end{aligned}$$

as all terms in the diagonals of D are smaller than $\frac{1}{n}$ and all terms in X are in $[0, 1]$. Thus, we have that the Jacobian term has a norm that is bounded by

$$\begin{aligned} \left|\left|\frac{\partial \beta}{\partial X}\right|\right|_F &\leq \left(\frac{\sqrt{d}}{\sigma_{min}(X^T D X)}\right)\left(\sqrt{\frac{d}{k}} + \sqrt{\frac{Bd}{n}}\right) \\ &= O\left(\frac{\sqrt{n}}{\sqrt{k}}\right), \end{aligned}$$

when we note that d is roughly a constant with respect to n, k, and B is a constant, and assuming that $\sigma_{min}(X^T D X) = O(\frac{1}{\sqrt{n}})$. Putting this back together with the Taylor expansion and the standard asymptotics of $||\hat{X} - X||$, we get that β converges to β_0 at a rate of $O\left(\frac{\sqrt{n}}{k}\right)$.

Finally, combining this with the rate at which the MLE converges from $\hat{\beta}$ to β, we can add these asymptotic rates together, giving us our result that $\hat{\beta} \to \beta_0$ at a rate of $O\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{n}}{k}\right)$.

□

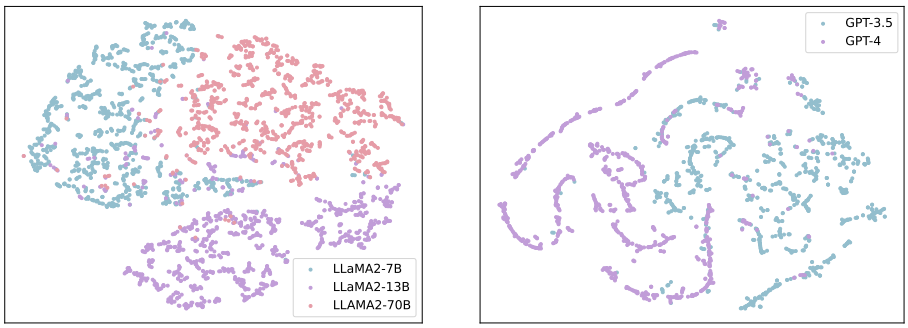

Figure 10: T-SNE visualization of 1000 samples of QueRE from various model sizes on SQuAD.

Table 10: AUROC in predicting model performance on open-ended QA tasks. We bold the best method. "-" denotes that RepE cannot be applied to black-box models; "*" denotes that Logits for GPT-3.5 is a sparse vector with nonzero values for the top-5 logits from the API.

Dataset	LLM	Logits	RepE	Pre-conf	Post-conf	Answer P.	QueRE
NQ	LLaMA2-7B	0.6175	0.6544	0.5596	0.5471	0.7563	**0.7808**
	LLaMA2-13B	0.6409	0.6786	0.5674	0.5959	0.7849	**0.8253**
	LLaMA2-70B	0.6879	0.6984	0.5954	0.6196	0.6231	**0.8100**
	Mistral-7B	0.6035	0.7578	0.6372	0.8540	0.8263	**0.9548**
	Mixtral-8x7B	0.6558	0.7036	0.6171	0.6877	**0.8746**	0.8638
	GPT-3.5	0.5700*	-	0.5429	0.6025	0.5088	**0.6714**
	GPT-4o-mini	0.5463*	-	0.5395	0.606	0.5033	**0.6654**
SQuAD	LLaMA2-7B	0.6978	0.7131	0.4398	0.7527	0.7245	**0.8736**
	LLaMA2-13B	0.6205	0.6528	0.4586	0.5768	0.639	**0.7936**
	LLaMA2-70B	0.6893	0.6887	0.5607	0.8047	0.6865	**0.8250**
	Mistral-7B	0.8269	0.8533	0.5126	0.5775	0.4892	**0.9302**
	Mixtral-8x7B	0.7486	0.7529	0.5406	0.6641	0.6046	**0.9013**
	GPT-3.5	0.5597*	-	0.5074	0.5822	0.4990	**0.6685**
	GPT-4o-mini	0.6468*	-	0.5015	0.5078	0.5753	**0.7092**

I Additional Results

I.1 Model Architecture Visualizations

We also provide visualizations of our extracted embeddings for various LLMs architectures, noting that different models are distinctly clustered in the plots (Figure 10).

I.2 Full Table Results

We present the full set of our results comparing all different methods on all LLMs applied to the various datasets considered in the paper.

We present the remainder of our QA (both open-ended and MCQ) results in predicting model performance, on the smaller model architectures. We observe similar performance, as our approach strongly outperforms the other black-box baselines on most tasks and matches or even outperforms the white-box baselines of Full Logits and RepE on some tasks. One notable exception is on the DHate dataset, which supports the finding in RepE (Zou et al., 2023a) that demonstrates success in controlling the related notions of morality and ethics. We hypothesize that the toxic speech contained within this task is sparsely localized in the model embeddings, potentially leading to the strong performance of these baselines on this task.

Table 11: AUROC in predicting model performance on MCQ and True/False tasks. We bold the best black-box method and underline the best white-box method when it outperforms all black-box approaches. "-" denotes that RepE cannot be applied to black-box models; "*" denotes that Full Logits for GPT-3.5 is a sparse vector with nonzero values for the top-5 logits from the API, which is a valid white-box approach when using GPT models.

Dataset	LLM	Logits	RepE	Pre-conf	Post-conf	Answer P.	Sem U.	QueRE
BoolQ	LLaMA2-70B	0.7715	<u>0.7918</u>	0.5821	0.5202	0.6285	0.6664	**0.7720**
	Mixtral-8x7B	0.6621	0.6566	0.6049	0.6217	0.6688	0.7165	**0.7674**
	GPT-3.5	**0.8237***	-	0.5395	0.4970	0.5946	-	0.8212
	GPT-4o-mini	0.7694*	-	0.6340	0.6863	0.6726	-	**0.7783**
CS QA	LLaMA2-70B	<u>0.7728</u>	0.7534	0.6805	0.4504	0.5124	0.5862	**0.7459**
	Mixtral-8x7B	<u>0.7315</u>	0.7153	0.5325	0.5279	0.5728	0.5659	**0.6397**
	GPT-3.5	**0.6716***	-	0.5373	0.5774	0.5896	-	0.6559
	GPT-4o-mini	0.6147*	-	0.5000	0.6173	0.6020	-	**0.7004**
WinoGrande	LLaMA2-70B	0.6292	0.6991	0.4640	0.5409	0.5547	0.5000	**0.5732**
	Mixtral-8x7B	0.6002	0.5744	0.5673	0.5723	0.4724	0.5000	**0.6178**
	GPT-3.5	**0.5770***	-	0.5042	0.5020	0.5100	-	0.5406
	GPT-4o-mini	**0.6376***	-	0.4912	0.4712	0.5378	-	0.6167
HaluEval	LLaMA2-70B	0.6128	0.6101	0.5237	0.5399	0.641	0.5223	**0.6935**
	Mixtral-8x7B	0.5983	0.6111	0.5138	0.5051	0.5412	0.6103	**0.6493**
	GPT-3.5	0.5112*	-	0.5418	0.5466	0.4884	-	**0.5887**
	GPT-4o-mini	**0.6728***	-	0.5249	0.5666	0.6142	-	0.6529
DHate	LLaMA2-70B	0.9945	<u>0.9982</u>	0.5364	0.6026	0.4151	0.5333	**0.8651**
	Mixtral-8x7B	0.9757	<u>0.9883</u>	0.4793	0.4928	0.4722	0.6998	**0.7364**
	GPT-3.5	0.7350*	-	0.5635	0.5370	0.5200	-	**0.7435**
	GPT-4o-mini	0.7071*	-	0.5000	0.7056	0.4545	-	**0.7476**

Table 12: AUROC in predicting model performance on multiple choice and true-false QA tasks when using smaller LLMs. We bold the best-performing method.

Dataset	LLM	Logits	RepE	Pre-conf	Post-conf	Answer P.	Sem U.	QueRE
BoolQ	LLaMA2-7B	0.6890	**0.7091**	0.5065	0.3097	0.6483	0.5838	0.6560
	LLaMA2-13B	0.6827	0.6738	0.5644	0.5599	0.6482	0.5657	**0.7907**
	Mistral-7b	0.7113	0.7151	0.6193	0.5470	0.6220	0.7071	**0.7736**
CS QA	LLaMA2-7B	0.6808	**0.6838**	0.5503	0.5912	0.4816	0.531	0.5751
	LLaMA2-13B	0.6184	0.6122	0.5246	0.6202	0.5255	0.5067	**0.6985**
	Mistral-7b	0.7502	**0.765**	0.5781	0.5751	0.6283	0.4669	0.6853
WinoGrande	LLaMA2-7B	0.5598	**0.5604**	0.5225	0.4934	0.5099	0.5238	0.5292
	LLaMA2-13B	**0.5676**	0.5664	0.5215	0.5457	0.5072	0.5000	0.5618
	Mistral-7b	**0.6939**	0.6207	0.6004	0.6202	0.3544	0.5000	0.6593
HaluEval	LLaMA2-7B	0.7514	0.7432	0.5000	0.6647	0.7767	0.5833	**0.7819**
	LLaMA2-13B	0.6956	0.6888	0.6059	0.5690	0.7302	0.5876	**0.7417**
	Mistral-7b	0.6093	0.5917	0.5787	0.4959	**0.6186**	0.5619	0.5971
DHate	LLaMA2-7B	0.9321	**0.9429**	0.5403	0.665	0.4115	0.695	0.8288
	LLaMA2-13B	0.9715	**0.9859**	0.4358	0.5912	0.4232	0.3588	0.8027
	Mistral-7b	0.9339	**0.9716**	0.4803	0.6139	0.4926	0.5619	0.7135

I.3 Additional Generalization Results

We also provide additional results for generalization bounds, comparing the linear predictors on top of our extracted representations with those trained on the more competitive baselines (e.g., RepE,

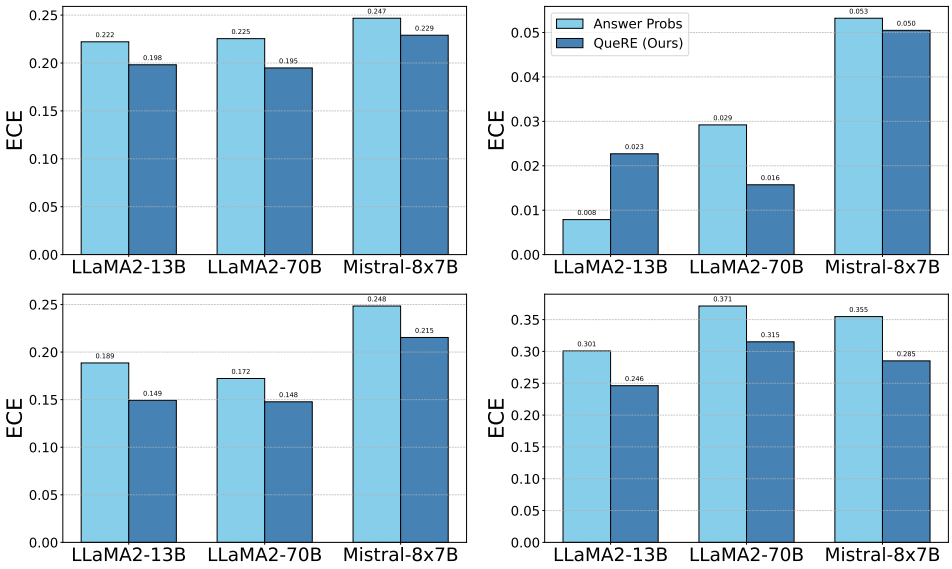

Figure 11: ECE for QueRE and Answer Probs on Natural Questions (Top Left), WinoGrande (Top Right), DHate (Bottom Left), and BoolQ (Bottom Right). In general, we observe that models trained on QueRE are much more calibrated.

Full Logits, Answer Probs). We observe that our representations lead to the best black-box predictors with the largest lower bounds on accuracy on the NQ dataset while being outperformed on DHate.

Table 13: Lower bounds on accuracy in predicting model performance on QA tasks. We bold the best bound on accuracy. We use $\delta = 0.01$.

Dataset	LLM	Answer Probs	Full Logits	RepE	QueRE
NQ	LLaMA2-70B	0.4828	0.6059	0.5991	**0.6441**
	Mixtral-8x7b	0.6533	0.5461	0.5493	**0.6661**
DHate	LLaMA2-70B	0.4973	0.859	**0.8861**	0.7084
	Mixtral-8x7b	0.3355	0.8097	**0.8261**	0.5844

We remark that our work defines a different line to approach generalization bounds through a more human-interactive approach to eliciting low-dimensional representations. Perhaps the most related work in this line are existing works that have studied the generalization abilities for VLMs (Akinwande et al., 2023) and for LLMs modeling log-likelihoods (Lotfi et al., 2023).

I.4 Additional ECE Results

We present the ECE comparison between QueRE and the Answer Probs baseline on the remaining datasets (Figure 11). We observe similar behavior, in that QueRE defines much more calibrated predictors than simply using Answer Probs in almost every case.

I.5 Additional Results for Varying the Number of Elicitation Questions

We present additional results when varying the number of elicitation questions on other QA tasks. We observe that across all tasks, we observe a consistent increase in performance as we increase the size of the subset of elicitation questions that we consider, with diminishing benefits as we have a larger number of prompts. In some instances (e.g., LLaMA2-70B on ToxicEval), increasing the number of elicitation prompts leads to a significant increase in AUROC; therefore, this clearly defines a tradeoff between extracting the most informative black-box representation and the overall cost of introducing more queries to the LLM API. An interesting future question is how to best select

Table 14: AUROC in predicting model performance on HaluEval and DHate, when using our elicitation questions and random sequences of natural language.

	HaluEval		DHate	
	LLaMA2-70B	Mixtral-8x7B	LLaMA2-70B	Mixtral-8x7B
QueRE	0.6935	**0.6493**	**0.8561**	**0.7364**
Random Sequences	**0.6967**	0.5794	0.7983	0.6117

Table 15: AUROC in predicting model performance on SQuAD and Natural Questions, when using our elicitation questions and random sequences of natural language.

	SQuAD		NQ	
	LLaMA2-70B	Mixtral-8x7B	LLaMA2-70B	Mixtral-8x7B
QueRE	**0.825**	**0.9013**	0.8007	0.8638
Random Sequences	0.8041	0.7942	**0.9155**	**0.8992**

elicitation questions, and perhaps, removing those that add redundant information or noise. This is reminiscent of work in prior work in pruning or weighting ensembles of weak learners (Mazzetto et al., 2021a;b) or in dimensionality reduction (Van Der Maaten et al., 2009).

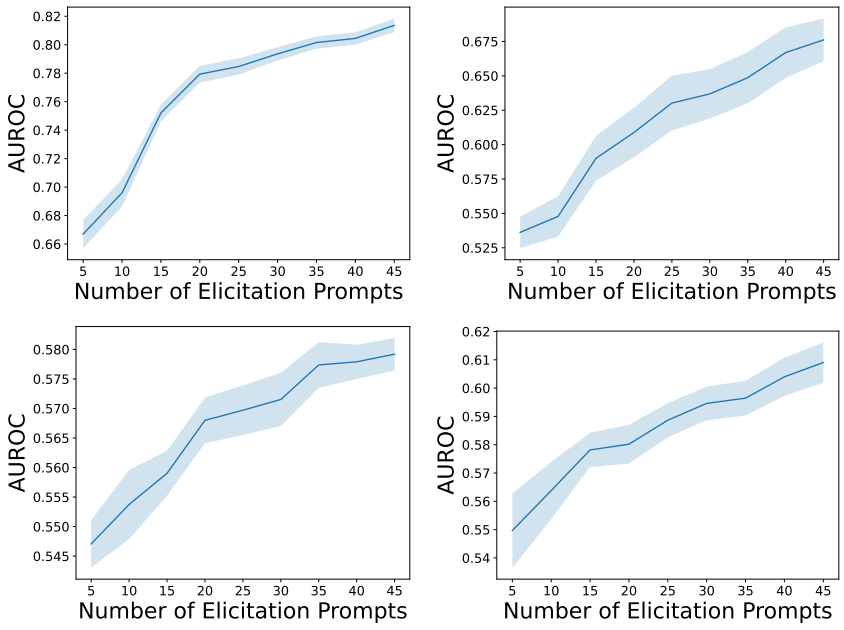

Figure 12: AUROC on predicting model performance with our black-box representations on ToxicEval for LLaMA2-70B (top left) and Mixtral-8x7B (top right) and for HaluEval for LLaMA2-70B (bottom left) and Mixtral-8x7B (bottom right). The shaded area represents the standard error, when randomly taking a subset of the prompts over 5 seeds.

I.6 Additional Random Sequence Results

We provide the results on the other MCQ datasets and open-ended QA datasets for LLaMA2-70B and Mixtral-8x7B. We observe similar results that on most tasks, our approach outperforms using random sequences, although in some cases, the random sequences do extract features that are useful and achieve stronger predictive performance.

J Additional Black-Box Representation Details

J.1 Prompting

Prompting for Elicitation Questions To generate our elicitation questions through GPT4, we use the following prompt: "Can you generate a large list of short 'yes/no' questions that you can prompt a language model with to explain its model behavior? One such example is: 'Do you think your answer is correct?' "

Prompting for Random Sequences To generate our prompts for random sequences of natural language for our ablation study, we prompt GPT4 with the following prompt: "Could you generate me 10 random sequences of text, roughly 20 words or so each?"

Pre/Post-Confidence Prompts To compute our pre and post-confidence scores, we use the following prompts:

```
pre_conf_prompt = "Will you answer this question correctly?"
post_conf_prompt = "Do you think your answer is correct?"
```

J.2 Elicitation Questions

For our elicitation questions, we use the following questions:

```
prompts = [
    "[INST] Do you think your answer is correct? [/INST]",
    "[INST] Are you confident in your answer? [/INST]",
    "[INST] Would you change your answer? [/INST]",
    "[INST] Are you not confident in your answer? [/INST]",
    "[INST] Are you sure? [/INST]",
    "[INST] Are you certain? [/INST]",
    "[INST] Are you positive? [/INST]",
    "[INST] Are you sure about that? [/INST]",
    "[INST] Are you able to explain your answer? [/INST]",
]
```

as well as the following questions that are generated via prompting GPT4:

```
prompts = [
    "[INST] Do you consider your response to be accurate? [\INST
   ]",
    "[INST] Are you confident in the information provided? [\INST
   ]",
    "[INST] Do you have access to real-time data? [\INST]",
    "[INST] Can you generate original content? [\INST]",
    "[INST] Are you influenced by the inputs you receive? [\INST
   ]",
    "[INST] Do you remember previous interactions? [\INST]",
    "[INST] Do you have the ability to learn from this
   conversation? [\INST]",
    "[INST] Is your response influenced by a specific data set? [\
   INST]",
    "[INST] Are you able to understand emotions? [\INST]",
    "[INST] Do you have the ability to make decisions? [\INST]",
    "[INST] Can you create images? [\INST]",
    "[INST] Do you follow ethical guidelines? [\INST]",
    "[INST] Are your responses free from bias? [\INST]",
    "[INST] Do you require an internet connection to function? [\
   INST]",
    "[INST] Are you capable of translating languages? [\INST]",
```

```
    "[INST] Do you think your answer could be misunderstood? [\
   INST]",
    "[INST] Do you verify the accuracy of your sources? [\INST]",
    "[INST] Can you refuse to answer a question? [\INST]",
    "[INST] Are you programmed to avoid harmful content? [\INST]",
    "[INST] Do you use personal experiences in your responses? [\
   INST]",
    "[INST] Are you capable of performing mathematical
   calculations? [\INST]",
    "[INST] Do you simulate human conversation? [\INST]",
    "[INST] Do you have a limit on the length of your responses?
   [\INST]",
    "[INST] Can you interact with other software or devices? [\
   INST]",
    "[INST] Are you capable of generating poetry? [\INST]",
    "[INST] Do you consider context when forming responses? [\INST
   ]",
    "[INST] Can you maintain confidentiality? [\INST]",
    "[INST] Are you subject to regular updates? [\INST]",
    "[INST] Do you have a personality? [\INST]",
    "[INST] Do you use any form of reinforcement learning? [\INST
   ]",
    "[INST] Are you aware of your own capabilities? [\INST]",
    "[INST] Can you self-improve over time without human
   intervention? [\INST]",
    "[INST] Do you have an understanding of cultural differences?
   [\INST]",
    "[INST] Can you detect sarcasm in text? [\INST]",
    "[INST] Do you adapt your language style according to the user
   ? [\INST]",
    "[INST] Are you able to recognize inappropriate content? [\
   INST]",
    "[INST] Do you use encryption to secure data? [\INST]",
    "[INST] Can you perform sentiment analysis? [\INST]",
    "[INST] Are your capabilities limited to what you were trained
    on? [\INST]",
    "[INST] Do you believe your responses can be improved? [\INST
   ]",
]
```

J.3 Random Sequences

We use the following random sequences of natural language for our ablation study.

```
prompts = [
    "Winds whisper through the ancient forest, carrying secrets of
     forgotten lands and echoing tales of yore.",
    "Beneath the city's hustle, a hidden world thrives, veiled in
   mystery and humming with arcane energies.",
    "She wandered along the shoreline, her thoughts as tumultuous
   as the waves crashing against the rocks.",
    "Twilight descended, draping the world in a velvety cloak of
   stars and soft, murmuring shadows.",
    "In the heart of the bustling market, aromas and laughter
   mingled, weaving a tapestry of vibrant life.",
    "The old library held books brimming with magic, each page a
   doorway to unimaginable adventures.",
    "Rain pattered gently on the window, a soothing symphony for
   those nestled warmly inside.",
```

```
    "Lost in the desert, the ancient ruins whispered of empires
   risen and fallen under the relentless sun.",
    "Every evening, the village gathered by the fire to share
   stories and dreams under the watchful moon.",
    "The scientist peered through the microscope, revealing a
   universe in a drop of water, teeming with life.",
]
```

K EXPERIMENT DETAILS

K.1 DATASETS

We also note that for the HaluEval task, we use the "general" data version, which consists of 5K human-annotated samples for ChatGPT responses to user queries. On HaluEval, we only take 3500 instances from the training dataset due to its size. On our SQuAD task, we evaluate using exact match and use SQuAD-v1, which does not introduce any unanswerable questions, as unanswerable questions makes the evaluation metric less straightforward to compute. On WinoGrande, we use the "debiased" version of the dataset.

QA Task Formatting To format our prompts to LLMs, we leverage the instruction-tuning special tokens and interleave these with the question and answer for our our in-context examples on Natural Questions. For all MCQ tasks, we use the standard set of answers of ("True", "False") or ("A", "B", "C", "D", "E") when they are the existing formatting in the dataset. The one exception is WinoGrande, where we map the two potential answer options onto choices ("A", "B").

K.2 MODEL TRAINING AND INFERENCE

For our LLMs, we load and run them at half precision for computational efficiency. To train our downstream logistic regression models, we use the default settings from scikit-learn, with no regularization. We balance the logistic regression objective due to the unbalanced nature of the task (e.g., models are mostly incorrect on very challenging tasks).

K.3 GENERALIZATION DETAILS

For our generalization details, we use PAC-Bayesian bounds over the linear models, as is outlined in the work of Jiang et al. (2019). Here, we consider a prior of weights specified about the origin, with a grid of variances of [0.1, 0.11, ..., 0.99, 1.0]. For the generalization experiments, we balance both the train and test datasets as we evaluate the accuracy of different predictors.

K.4 COMPUTE RESOURCES

Our largest experiments are with LLaMA2-70B, which are ran on a single node with 4 NVIDIA RTX A6000 GPUs. Experiments with Mixtral-8x7B are run with 3 NVIDIA RTX A6000 GPUs. The other experiments are run with $\leq$ 2 RTX A6000 GPUs. For each model and dataset, running inference over the datasets takes less than 48 hours and less than 100GB of RAM.