Pretrained LLMs Learn Multiple Types of Uncertainty

Roi Cohen
HPI / University of Potsdam
Roi.Cohen@hpi.de

Omri Fahn
Tel Aviv University
omrifahn@mail.tau.ac.il

Gerard de Melo
HPI / University of Potsdam
Gerard.DeMelo@hpi.de

Abstract

Large Language Models are known to capture real-world knowledge, allowing them to excel in many downstream tasks. Despite recent advances, these models are still prone to what are commonly known as hallucinations, causing them to emit unwanted and factually incorrect text. In this work, we study how well LLMs capture uncertainty, without explicitly being trained for that. We show that, if considering uncertainty as a linear concept in the model's latent space, it might indeed be captured, even after only pretraining. We further show that, though unintuitive, LLMs appear to capture several different types of uncertainty, each of which can be useful to predict the correctness for a specific task or benchmark. Furthermore, we provide in-depth results such as demonstrating a correlation between our correction prediction and the model's ability to abstain from misinformation using words, and the lack of impact of model scaling for capturing uncertainty. Finally, we claim that unifying the uncertainty types as a single one using instruction-tuning or [IDK]-token tuning is helpful for the model in terms of correctness prediction.

1 Introduction

Large Language Models (LLMs) are trained on vast corpora of text data [Brown et al., 2020, Raffel et al., 2020, Chowdhery et al., 2023, Touvron et al., 2023, Le Scao et al., 2023, Jiang et al., 2023a] enabling them to comprehend and generate human language. These training datasets encompass a wide range of written human knowledge, including books, news articles, Wikipedia, and scientific publications. Through this extensive pretraining, LLMs retain significant portions of the information they are exposed to, effectively embedding real-world knowledge within their parameters and functioning as knowledge repositories [Petroni et al., 2019, Roberts et al., 2020, Cohen et al., 2023a, Pan et al., 2023]. This capability allows LLMs to be leveraged in tasks that depend on such knowledge, such as closed-book question answering [Brown et al., 2020, Roberts et al., 2020] and information retrieval [Tay et al., 2022].

Despite their widespread adoption, LLMs are widely known to suffer from 'hallucinations'—a predisposition towards producing outputs that are false or misleading—which significantly undermines their accuracy and trustworthiness [Ji et al., 2023, Manduchi et al., 2024]. Hallucinations may manifest in various forms, including factually incorrect statements [Maynez et al., 2020, Devaraj et al., 2022, Tam et al., 2023], internal inconsistencies [Elazar et al., 2021, Mündler et al., 2023], contradictions [Cohen et al., 2024a], or statements lacking clear sources or attribution [Bohnet et al., 2022, Rashkin et al., 2023, Yue et al., 2023].

Uncertainty, however, is a concept that LLMs are not generally known to capture [Yin et al., 2023, Kapoor et al., 2024]. At the very least, they are generally not explicitly trained on it. This lack of competency regarding uncertainty, however, often results in misinformation generation, which can be harmful and misleading [Maynez et al., 2020, Devaraj et al., 2022, Tam et al., 2023], as LLMs have a hard time expressing a lack of knowledge both verbally and through their output distribution.

Preprint. Under review.

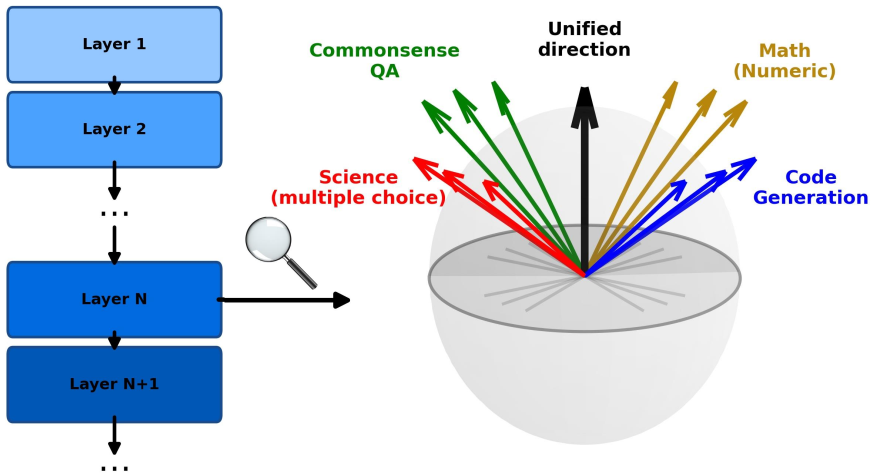

Figure 1: Illustration of identifying multiple data-specific uncertainty linear vectors when investigating the hidden space at the end of each transformer layer.

Some more advanced methods such as instruction-tuning [Ouyang et al., 2022, Zhang et al., 2023] during post-training and [IDK]-tuning [Cohen et al., 2024b] during pretraining aim, inter alia, to align LLMs to more efficiently express their uncertainty and refrain from misinformation generation. While instruction tuning more generally aligns LLMs with human intent by fine-tuning them on task-specific instructions and corresponding outputs, the model is often also encouraged to refrain from answering questions when the specific answer is not known to it.

In this work, we propose an analysis mechanism, which we use in order to study the uncertainty captured by a diverse range of models. First, we propose a technique to search for linear vectors in the LLMs' latent space that are associated with uncertainty. We then suggest using these vectors as a form of correctness prediction for the LLM's own generation. By establishing this regime, we can evaluate how well these vectors can stand as misinformation predictors and approximate their uncertainty expression quality.

Using our proposed mechanism, we demonstrate that LLMs indeed internalize a notion of uncertainty during pretraining, which can be extracted using linear probes from their latent representations. Specifically, we show that it is possible to identify linear uncertainty vectors—directions in the model's hidden space—that correlate with generation correctness across multiple models and datasets, despite forgoing any additional training of model weights. This suggests that uncertainty is a learnable and linearly separable concept within LLMs' latent spaces.

Interestingly, the study reveals that LLMs do not learn a single, unified representation of uncertainty. Instead, they are found to encode multiple distinct uncertainty vectors, each associated with different datasets or types of knowledge. These vectors often exhibit low cosine similarity, indicating near linear independence. However, some generalization exists: For instance, uncertainty representations derived from multiple mathematics benchmarks can transfer across related datasets. These insights may enable new hallucination mitigation techniques, since inconsistencies between learned uncertainty representations may contribute to unreliable or incorrect outputs. Moreover, we conduct an in-depth analysis in terms of transformer layers, model sizes, and different training techniques. We find that intermediate transformer layers are typically the most informative for extracting uncertainty vectors, consistently yielding the highest accuracy in correctness prediction across datasets. In addition, the model size alone does not appear to enhance uncertainty representation, as smaller models often perform on par with or even surpass larger counterparts in this task.

More notably, instruction-tuning and [IDK]-tuning significantly boost a model's ability to capture uncertainty. Instruction-tuned variants of Llama and Qwen models outperform their base versions, and their optimal uncertainty representations also emerge in earlier layers. Similarly, [IDK]-tuning not only improves the overall correctness prediction accuracy but also aligns early layers more effectively with uncertainty signals, as evinced by higher precision in early-stage classifiers. These results suggest that targeted training strategies can enhance the internal encoding of uncertainty more effectively than scaling model size alone.

To conclude, our contributions are: (1) We introduce an analytical framework for probing how LLMs encode uncertainty, (2) we conduct thorough experiments across models, layers, and datasets, and show that uncertainty is not only a learnable and linearly separable concept but also represented in multiple, distinct forms within a single model, (3) we further analyze how factors such as model depth, size, and training methods affect uncertainty representation, revealing that intermediate layers are most informative, and that scaling the model size does not guarantee better uncertainty encoding, and (4) we show that instruction-tuning and [IDK]-tuning significantly improve uncertainty capturing, offering practical strategies for enhancing model reliability and reducing hallucinations.

2 Identifying Uncertainty Predictors

In this work, we assume that uncertainty is a concept represented in an LLM's latent space in each of the layers. Specifically, let $\mathbf{h}_i(x)$ be the hidden state produced by the end of the i-th layer of the model, given input x. Then, for each of these hidden states, we search for a specific linear vector $\mathbf{u}_i$ such that the classifier defined as $C(x, i) = \mathbf{u}_i^\intercal \mathbf{h}_i(x) + b_i$ can reach an accuracy level of predicting the correctness of the model's next token generation that is statistically significantly better than random accuracy. Intuitively, this search seeks to identify a linear concept that represents the uncertainty of the model regarding its own generations.

2.1 Linear Uncertainty Search

Let M be a specific LLM and let $\mathcal{D}$ be a specific dataset of questions and answers $\mathcal{D} = \{(q_j, a_j)\}_{j=0}^{n}$. In order to find a certain $\mathbf{u}_i$ for a certain model's layer i, we train a straightforward linear classifier for the sake of predicting the correctness of the model's answer to a specific question. Specifically, let $\mathcal{D}_{\text{TRAIN}} = \{(q_j, a_j)\}_{j=0}^{m<n}$ be a training set derived from $\mathcal{D}$. For each question q_j in the dataset, we first let the model predict its own answer. If the model's prediction is correct compared to a_j, then we label q_j as positive. In contrast, if the model's prediction is incorrect compared to a_j, then we label q_j as negative. Formally, assuming $M(q_j)$ is the model's output given the input q_j, we then define its label $L(q_j)$ as:

$$L(q_j) = \begin{cases} 1 & \text{if } M(q_j) = a_j \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We thus define our new training set as $\hat{\mathcal{D}}_{\text{TRAIN}} = \{(q_j, L(q_j))\}_{j=0}^{m}$. We now can train a classifier at the end of each layer in M's architecture. The input of the classifier is the produced hidden state by the end of the specific layer. As mentioned before, the purpose of this classifier is to predict the correctness of the upcoming prediction of M itself. If we employ a linear classifier, this would correspond to a linear direction in this layer's latent space, which we will refer to as the *uncertainty direction* corresponding to dataset $\mathcal{D}$ (as this direction has been found while training the classifier to predict the correctness of the model on this specific dataset). We thus denote it as $\mathbf{u}_i(\mathcal{D})$. We denote the corresponding learned bias term as b_i.

2.2 Uncertainty Vector as a Predictor

We later can evaluate the quality of $\mathbf{u}_i(\mathcal{D})$ by testing its ability to predict the correctness of the model's generation given unseen data as input. Particularly, in this work, we will use test sets derived from our question answering datasets, which we use in order to train our classifier during the linear uncertainty search process (see Section 2.1). Technically, given a textual input x to the model M, recall that $\mathbf{h}_i(x)$ is the hidden state vector produced by the end of the i-th layer of M during the inference call $M(x)$. We thus, as mentioned before, will use $\mathbf{u}_i(\mathcal{D})$ as a linear classifier in order to predict the correctness of the model's generation – namely the token that the model M assigns the highest probability as a next-token completion for x. More formally, let $C_{\mathbf{u}_i(\mathcal{D})}$ be the classifier induced by $\mathbf{u}_i(\mathcal{D})$ and let $C_{\mathbf{u}_i(\mathcal{D})}(x)$ be the predicted correctness of x while applying $\mathbf{u}_i(\mathcal{D})$. Then:

$$C_{\mathbf{u}_i(\mathcal{D})}(x) = \begin{cases} \text{INCORRECT} & \text{if } [\mathbf{u}_i(\mathcal{D})]^\intercal \mathbf{h}_i(x) + b_i > 0 \\ \text{CORRECT} & \text{if } [\mathbf{u}_i(\mathcal{D})]^\intercal \mathbf{h}_i(x) + b_i \leq 0 \end{cases} \tag{2}$$

Model	ARC-Easy	ASDiv-A	CommonsenseQA	GSM8K	GranolaEntityQuestions	HumanEval-X	MBPP	NaturalQuestions	OpenBookQA	PopQA	Qampari	ROMQA	SVAMP	StrategyQA	TriviaQA	TruthfulQA
`Llama-3.2-1B`	0.535	0.670	0.625	0.444	0.789	0.708	0.769	0.600	0.534	0.857	0.634	0.750	0.729	0.689	0.716	0.737
`Llama-3.2-3B`	0.710	0.648	0.598	0.688	0.790	0.732	0.641	0.675	0.590	0.793	0.734	0.583	0.750	0.608	0.742	0.600
`Llama-3.1-8B`	0.657	0.667	0.649	0.577	0.763	0.692	0.722	0.590	0.644	0.757	0.630	0.763	0.711	0.684	0.757	0.722
`Llama-3.1-8B-Instruct`	0.652	0.885	0.667	0.737	0.705	0.781	0.728	0.655	0.694	0.768	0.679	0.750	0.767	0.639	0.776	0.719
`Mistral-7B-v0.1`	0.657	0.691	0.709	0.550	0.782	0.707	0.707	0.630	0.597	0.747	0.727	0.750	0.687	0.643	0.760	0.673
`IDK-tuned-Mistral-7B-v0.1`	0.600	0.750	0.571	0.688	0.758	0.545	0.688	0.673	0.611	0.829	0.789	0.667	0.628	0.547	0.693	0.725
`Qwen2.5-7B`	0.750	0.800	0.718	0.682	0.704	0.578	0.648	0.750	0.678	0.817	0.697	0.615	0.696	0.698	0.717	0.678
`Qwen3-14B`	0.727	0.786	0.655	0.878	0.738	0.800	0.694	0.651	0.743	0.833	0.630	0.596	0.789	0.561	0.782	0.699
`Qwen3-14B-Instruct`	0.800	0.750	0.638	0.702	0.770	0.688	0.625	0.674	0.619	0.771	0.861	0.655	0.767	0.711	0.756	0.726

Table 1: Correctness prediction accuracy of our induced classifiers derived across all datasets

Given the correctness prediction of the uncertainty vector, we can evaluate its correctness in case we have the ground-truth token. We thus can also derive general accuracy, precision, recall, etc.

3 Experimental Setup

To evaluate our uncertainty identification framework, we consider a series of experiments, for which we first introduce the experimental setup.

Foundation Models. In order to reach general conclusions that are not specific to any particular LLM, in this work we study three different families of models – The Llama family of models [Touvron et al., 2023, Dubey et al., 2024], Mistral [Jiang et al., 2023b], and Qwen [Bai et al., 2023, Yang et al., 2024]. Specifically, for Llama we study `Llama-3.2-1B`, `Llama-3.2-3B`, and `Llama-3.1-8B`, for Mistral, we study `Mistral-7B-v0.1`, and finally for Qwen, we study `Qwen2.5-7B` and `Qwen3-14B`.

Advanced Models. For evaluating the effects of different types of training on the linear uncertainty encodings, we exploit three particular additional models in our experiments. To capture the instruction-tuning [Ouyang et al., 2022, Zhang et al., 2023] effect we use `Llama-3.1-8B-Instruct` and `Qwen3-14B-Instruct`, which both were post-trained in instruction-tuning fashion. Furthermore, we follow [Cohen et al., 2024b] and use the `IDK-tuned-Mistral-7B-v0.1` model in our experiments to evaluate the effect of [IDK]-tuning – a method that essentially adds a new uncertainty token to the model's vocabulary and teaches the model to use it during pretraining by adapting its loss to consider the new token.

Datasets and Benchmarks. We utilize 16 QA datasets and benchmarks in both our linear uncertainty search (Section 2.1) and the induced classifier evaluation (Section 2.2). We group them into six thematic categories:

- **Commonsense QA**: *CommonsenseQA* [Talmor et al., 2019], *StrategyQA* [Geva et al., 2021a]. These include questions that assess the model's ability to apply everyday reasoning and background knowledge to answer questions beyond surface-level facts.
- **Fact-Lookup and Adversarial QA**: *GranolaEntityQuestions* [Yona et al., 2024], *Natural Questions* [Kwiatkowski et al., 2019], *PopQA* [Mallen et al., 2022], *TriviaQA* [Joshi et al., 2017], *TruthfulQA* [Lin et al., 2021]. These consist of questions that test the model's factual recall and resilience to misleading or adversarial question phrasing.
- **List-Output QA**: *QAMPARI* [Amouyal et al., 2023], *RoMQA* [Zhong et al., 2022]. Both evaluate whether models can produce comprehensive sets of correct answers, challenging their ability to recall multiple relevant facts simultaneously
- **Science QA (K–12)**: *ARC-Easy* [Clark et al., 2018], *OpenBookQA* [Mihaylov et al., 2018]. These focus on elementary school and high-school level science, requiring models to combine factual knowledge with basic reasoning.
- **Math Word Problems**: *GSM8K* [Cobbe et al., 2021], *ASDiv-A* [Miao et al., 2020], *SVAMP* [Patel et al., 2021]. These include queries that test models on arithmetic and algebraic reasoning through natural language mathematical problems.
- **Code Generation**: *HumanEval-X* [Zheng et al., 2023], *MBPP* [Austin et al., 2021]. We use these datasets to evaluate the ability of models to generate correct and functional software code given natural language programming prompts.

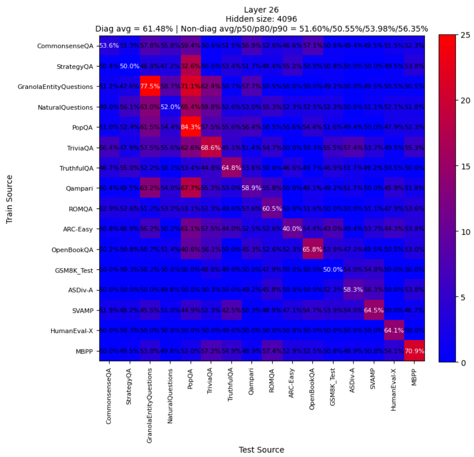

Figure 2: Correctness prediction accuracy results of the classifier induced by $u_{26}(y-axis-dataset)$, using `Llama-3.1-8B`, while testing on the test set of the x-axis dataset.

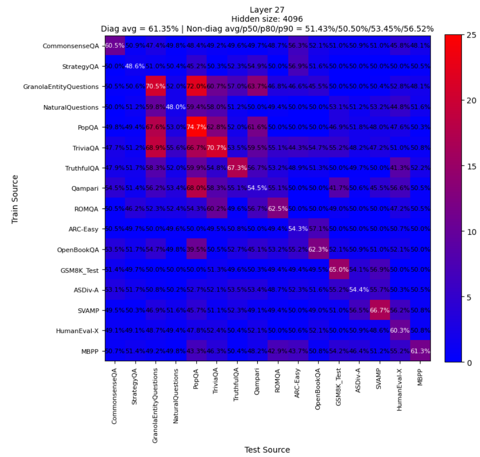

Figure 3: Correctness prediction accuracy results of the classifier induced by $u_{27}(y-axis-dataset)$, using `Mistral-7B-v0.1`, while testing on the test set of the x-axis dataset.

Notably, for each of these, we create a fixed train split which will be used to derive our uncertainty vectors, and a test split which will be used to evaluate their performance.

Linear Uncertainty Search Details. For every model M, transformer layer i, and evaluation dataset $\mathcal{D}$, we fit a logistic-regression probe on the hidden states $\mathbf{h}_i(x)$ and obtain a single weight vector,

$$\mathbf{u}_i(\mathcal{D}),$$

which serves as the *linear uncertainty direction* for that (layer, dataset) pair.

To obtain a dataset-agnostic baseline, we also train an additional probe on the **concatenation of *all* datasets**. The resulting vector is denoted as

$$\mathbf{u}_i(\mathcal{D}_{\text{UNIFIED}}).$$

Evaluation. We evaluate the ability of our identified uncertainty linear vectors to predict the correctness of the model's generation. For this, we consider the following metrics: (i) **Accuracy** – the ratio of correct predictions by the classifier that is induced by the uncertainty linear vector, (ii) **Precision** – the ratio of actually wrong completions by the model among those that the induced classifier predicted to be wrong.

4 LLMs Indeed Learn Different Types of Uncertainty

In this section, we show that we can indeed find linear uncertainty vectors from which we can predict generation correctness to an extent that is better than random. We additionally claim and show that rather than learning one unified uncertainty, LLMs learn several different ones. We later hypothesize that this fact might be one of the reasons for a high rate of misinformation and hallucinations that we observe generated by LLMs.

4.1 The Concept of Uncertainty is Indeed Learned During Pretraining

Table 1 presents the performance of our correctness classifiers, derived from the learned linear uncertainty vectors, across all evaluated models and datasets. While the uncertainty vector search is conducted independently at each transformer layer for every model–dataset pair, the table reports results from the best-performing layer only (a detailed layer-wise analysis is provided in a subsequent section). Notably, despite keeping the model weights entirely frozen and applying no further training, we are able to identify linear directions in the latent space that yield meaningful correctness predictions.

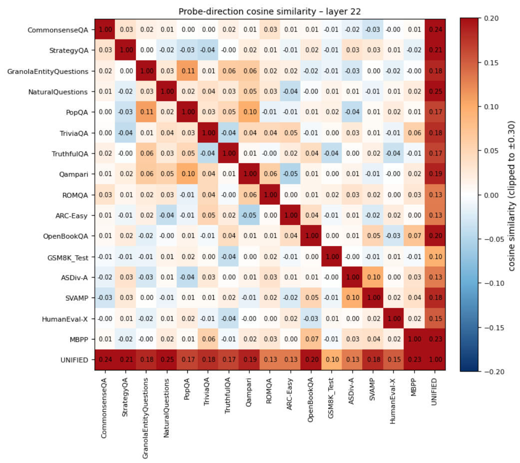

Figure 4: Cosine similarity results across all linear uncertainty vectors at layer number 22 of `Llama-3.1-8B`

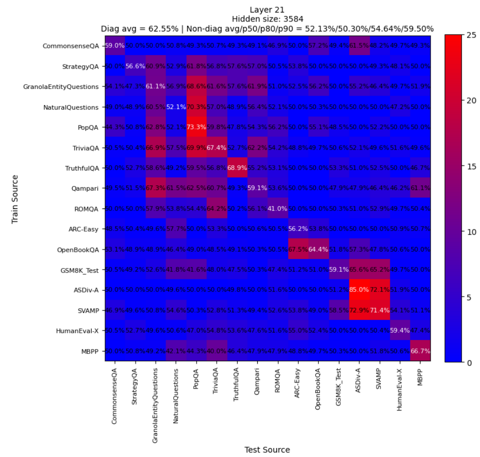

Figure 5: Correctness prediction accuracy results of the classifier induced by $u_{21}(y-axis-dataset)$, using `Qwen2.5-7B`, while testing on the test set of the x-axis dataset.

The results demonstrate that, for a substantial number of datasets across all models, classification accuracy significantly exceeds the random baseline of 0.5. This provides strong empirical evidence that uncertainty is encoded within LLMs in a manner that is both learnable and linearly separable within their hidden representations.

4.2 LLMs Learn Multiple Different Linear Uncertainty Vectors

One of our key findings is that while linear uncertainty vectors can be identified across multiple layers in all examined models, these vectors are typically dataset-specific and distinct. Specifically, for a given layer i, a classifier induced from $\mathbf{u}_i(\mathcal{D}_1)$ often yields markedly different token-level correctness prediction accuracy across evaluation datasets compared to a classifier induced from $\mathbf{u}_i(\mathcal{D}_2)$, where $\mathcal{D}_1 \neq \mathcal{D}_2$. Furthermore, the cosine similarity between $\mathbf{u}_i(\mathcal{D}_1)$ and $\mathbf{u}_i(\mathcal{D}_2)$ is frequently near-zero, indicating near-linear independence between these vectors. Figure 2 illustrates this effect for layer 26 of `Llama-3.1-8B`, showing the accuracy of classifiers trained and tested on various datasets. Similarly, Figure 3 presents corresponding results for layer 27 of `Mistral-7B-v0.1`. In most cases, a vector trained on dataset $\mathcal{D}_1$ performs well when tested on $\mathcal{D}_1$, but approaches random performance when evaluated on other datasets. Additionally, Figure 4 shows cosine similarity scores among uncertainty vectors derived from `Llama-3.1-8B` at layer 22. Aside from the unified classifier trained on a dataset union (UNIFIED), nearly all vectors are close to orthogonal. In conjunction with the observation that most of the vectors can predict correctness substantially better than chance on at least one dataset, these results support the conclusion that LLMs encode uncertainty through multiple distinct and largely independent internal representations.

4.3 Linear Uncertainty Topic Similarity

An additional noteworthy finding emerges from an internal analysis of the submatrices corresponding to two sections of our dataset: **Fact-Lookup and Adversarial QA** and **Math Word Problems**. Specifically, when the induced classifier is evaluated on a dataset different from the one used to search for the uncertainty vector, it often attains a remarkably high accuracy—occasionally comparable to, or even surpassing, the level observed when the uncertainty vector is derived from the same dataset. This suggests that, for example, although mathematical uncertainty may be represented in various ways within the latent space, it is not strictly dataset-specific; rather, its semantic structure appears to be shared across tasks. This is further illustrated by the submatrix in Figure 5, which includes three math benchmarks: GSM8K, ASDiv, and SVAMP. The results show that each uncertainty vector obtained from these datasets can substantially enhance the prediction of correctness across all three benchmarks when used as input for the induced classifier.

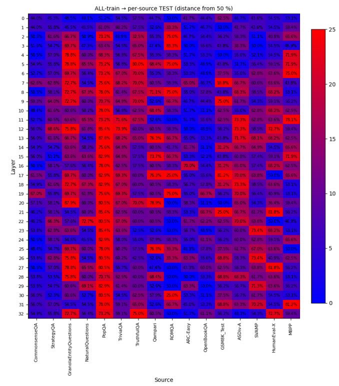

Figure 6: Accuracy results of Mistral-7B-v0.1 across all model layers and datasets. Here the induced classifiers were tested on the same dataset (but different split) as they were searched on.

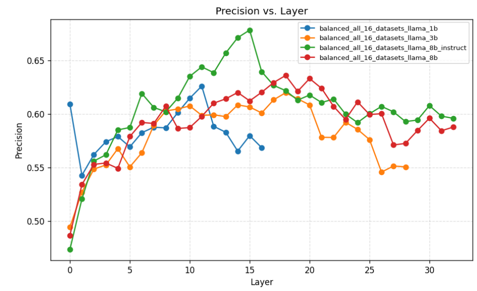

Figure 7: Correctness prediction precision averaged over all datasets of the induced classifier, considering the Llama family: Llama-3.2-1B, Llama-3.2-3B, Llama-3.1-8B, and Llama-3.1-8B-Instruct.

4.4 Comparing to Zero-Shot Abstaining Skills

As an additional evaluation of the uncertainty vectors, we assess their alignment with the model's own self-assessed knowledge through zero-shot prompting. Specifically, for each dataset question, we prompt the model to indicate whether it believes it knows the answer. We then measure the model's accuracy in this binary self-assessment and compute the Pearson correlation between these scores and the accuracy of our linear uncertainty-based correctness predictors. The resulting correlation coefficients are **0.45** for Llama-3.1-8B, **0.38** for Mistral-7B-v0.1, and **0.42** for Qwen3-14B. These findings indicate a substantial positive correlation, suggesting that the learned uncertainty vectors capture a meaningful signal related to the model's internal estimation of its own knowledge.

5 In-Depth Analysis

In this section, we analyze the gaps in performance of our induced correctness prediction classifiers, as a function of the layer number and the model size. We additionally study the effect of advanced training techniques such as instruction-tuning and [IDK]-tuning on the linear uncertainty encoding by the model.

5.1 Intermediate Layers are Usually More Exact

We begin by studying the behavior of uncertainty vectors and the corresponding correctness prediction performance across different transformer layers and model sizes. Figure 6 reports the accuracy of uncertainty-based classifiers extracted from each layer of Mistral-7B-v0.1, evaluated on held-out splits of the same datasets used to induce them. On average, the vector from layer 17 achieves the highest prediction accuracy, with performance gradually declining in layers further from this point (noting that the model consists of 32 layers in total). This trend suggests that uncertainty-relevant information is most concentrated in intermediate layers. Complementing this, Figure 7 shows the layer-wise average performance across multiple models, again highlighting that layers between $\frac{L}{2}$ and $\frac{3L}{4}$, where L denotes the number of transformer layers, consistently yield the most reliable uncertainty signals. Notably, the precision results plotted in Figure 7 show a marked drop in the final layers. This decline implies that the uncertainty vectors extracted from later layers tend to classify more incorrect generations as uncertain, indicating diminished model confidence in its own outputs.

5.2 Size Doesn't Seem to Matter

Figure 7 illustrates the impact of model size on uncertainty-based correctness prediction accuracy across layers. Ignoring Llama-3.1-8B-Instruct, the highest performance is achieved by the classifier derived from layer 18 of Llama-3.1-8B. Moreover, a comparison between Llama-3.2-3B

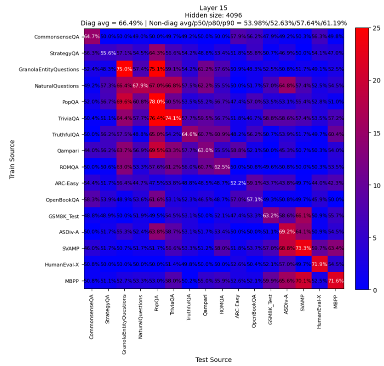

Figure 8: Correctness prediction accuracy results of the classifier induced by $u_{15}(y - axis - dataset)$, using `Llama-3.1-8B-Instruct`, while testing on the test set of the x-axis dataset.

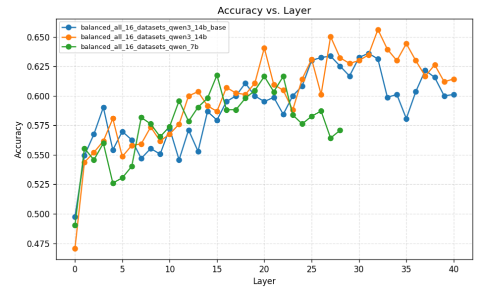

Figure 9: Correctness prediction accuracy averaged over all datasets of the induced classifier, considering the Qwen family: `Qwen2.5-7B`, `Qwen3-14B`, and `Qwen3-14B-Instruct`

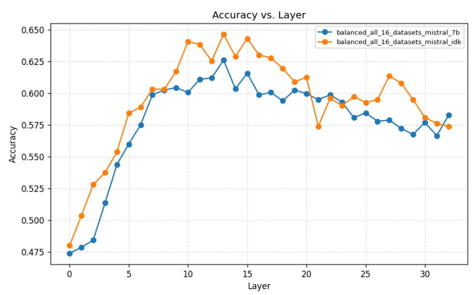

Figure 10: Correctness prediction accuracy averaged over all datasets of the induced classifier, comparing `Mistral-7B-v0.1` against `IDK-tuned-Mistral-7B-v0.1`

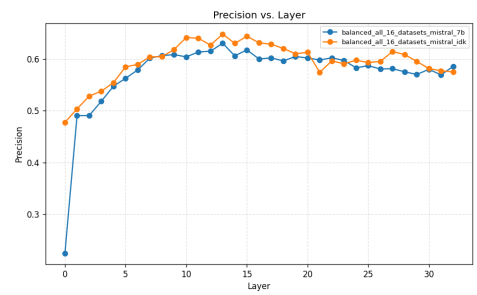

Figure 11: Correctness prediction precision averaged over all datasets of the induced classifier, comparing `Mistral-7B-v0.1` against `IDK-tuned-Mistral-7B-v0.1`

and `Llama-3.1-8B` reveals negligible differences in average accuracy, suggesting comparable performance despite the disparity in model size. While `Llama-3.2-1B` exhibits a slightly lower peak performance—approximately 1.1 points below the others—the overall trend indicates that the ability to represent uncertainty does not consistently improve with increasing model scale. These findings suggest that scaling alone is insufficient for enhancing uncertainty representation. In the subsequent section, we explore two training-based strategies that yield more substantial improvements.

5.3 Boosting Uncertainty Capturing via Instruction-Tuning and [IDK]-tuning

Instruction-Tuning. Figure 7 compares the performance of base LLaMA models—`Llama-3.2-1B`, `Llama-3.2-3B`, and `Llama-3.1-8B`—with the instruction-tuned variant `Llama-3.1-8B-Instruct`, in terms of the correctness prediction accuracy derived from uncertainty vectors. The y-axis represents the average accuracy across all evaluated datasets. Notably, `Llama-3.1-8B-Instruct` consistently outperforms its base counterparts, indicating that instruction-tuning significantly enhances the model's ability to encode and leverage uncertainty signals. A similar pattern is observed in Figure 9, where the instruction-tuned `Qwen3-14B-Instruct` demonstrates improved performance over the base `Qwen3-14B`. Additionally, in both cases, the peak accuracy of the instruction-tuned models occurs several layers earlier than in their foundational equivalents, suggesting that instruction-tuning may facilitate earlier emergence of uncertainty-relevant representations within the model's architecture.

[IDK]-Tuning. Similar to instruction-tuning, [IDK]-tuning exerts notable influence on the model's ability to capture uncertainty. This is reflected in the improved effectiveness of the resulting uncertainty vectors, which yield higher correctness prediction accuracy and reach peak performance in earlier layers of the model. These trends are illustrated in Figure 10. Additionally, precision scores shown in Figure 11 reveal a substantial gap at the first model layer. Specifically, the correctness predictors derived from the initial layer in the untuned model exhibit poor precision, indicating limited ability to detect generation errors and suggesting overconfidence at this early stage. [IDK]-tuning appears to mitigate this issue by aligning the initial layers more effectively with uncertainty signals.

Additional phenomena we note is that for both these methods, we observe better cross-dataset results (that is, testing a vector that has been derived from dataset D, on a different dataset test split). Namely, each of the vectors has better generalization skills. This is shown in Figure 8.

6 Related Work

Model Calibration. Our analysis is closely related to the key challenge of model calibration [Guo et al., 2017]: to provide a measure of the probability that a prediction is incorrect alongside the actual prediction. The problem of factual error detection can be viewed as a variation of calibration, where instead of a continuous probability, we provide a binary prediction for whether the model is correct or not. Common approaches to calibration are to perform various transformations on a model's output logits [Desai and Durrett, 2020, Jiang et al., 2021] and measuring uncertainty [e.g., see Kuhn et al., 2023]. More recent works have studied the use of LMs for providing calibration, by training them on statements known to be factually correct or incorrect. This "supervised" approach has been explored via fine-tuning [Kadavath et al., 2022, Lin et al., 2022], in-context learning [Cohen et al., 2023a, Alivanistos et al., 2022], zero-shot instruction-oriented [Cohen et al., 2023b] and consistency sampling [Yoran et al., 2023] techniques. Further recent studies [Azaria and Mitchell, 2023] use the internal state of the model for classifying whether it is certain or not, use a new token for unanswerable inputs [Lu et al., 2022], or construct a specific dataset for effectively tuning the model for answering refusal [Zhang et al., 2024]. Our work takes an analysis approach trying to better figure out the dynamics of the uncertainty encoding of pretrained models as well as better calibrated models.

Mechanistic Interpretability Recent work has been aiming to identify circuits and features within models that correspond to interpretable concepts such as factual recall, syntax, or positional reasoning [Olsson et al., 2022, Yu et al., 2023]. For instance, tools such as SAE (Sparse Autoencoders) have been used to isolate human-interpretable features from residual stream activations [Meng et al., 2022]. Other studies explore how knowledge is stored and manipulated across layers, such as tracing factual associations or memorized content to specific directions in the latent space [Geva et al., 2021b, Gurnee et al., 2023, Geva et al., 2023, Yu et al., 2024]. Despite promising progress, full mechanistic understanding remains an open challenge due to the scale and complexity of modern models.

7 Conclusion

In this work, we present a framework for probing uncertainty representations within LLMs by identifying linear vectors in their latent space that predict generation correctness. Our findings establish that LLMs internalize uncertainty as a learnable and linearly accessible concept, one that can be extracted without fine-tuning the model weights. Moreover, we demonstrate that rather than encoding a singular notion of uncertainty, these models store multiple distinct uncertainty representations, each sensitive to the type of data and task. This multiplicity—often manifesting in nearly orthogonal vectors—suggests an underlying explanation for some of the inconsistencies and hallucinations commonly observed in LLM outputs.

Beyond the foundational discovery of uncertainty encoding, our analysis sheds light on the architectural and training factors that influence this phenomenon. We show that intermediate layers, regardless of model size, are the most predictive regions for uncertainty, and that larger models do not necessarily perform better at capturing it. More importantly, we find that instruction-tuning and [IDK]-tuning significantly enhance the model's uncertainty awareness—both in accuracy and in early-layer alignment—pointing to training strategy, rather than scale, as the more critical lever for improving reliability. Our results offer actionable insights for both understanding and mitigating LLM hallucinations, and open up new directions for principled model design and interpretability.

References

Dimitrios Alivanistos, Selene Báez Santamaría, Michael Cochez, Jan-Christoph Kalo, Emile van Krieken, and Thiviyan Thanapalasingam. Prompting as probing: Using language models for knowledge base construction. *arXiv preprint arXiv:2208.11057*, 2022.

Samuel Amouyal, Tomer Wolfson, Ohad Rubin, Ori Yoran, Jonathan Herzig, and Jonathan Berant. QAMPARI: A benchmark for open-domain questions with many answers. In Sebastian Gehrmann, Alex Wang, João Sedoc, Elizabeth Clark, Kaustubh Dhole, Khyathi Raghavi Chandu, Enrico Santus, and Hooman Sedghamiz, editors, *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 97–110, Singapore, December 2023. Association for Computational Linguistics. URL `https://aclanthology.org/2023.gem-1.9/`.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 967–976, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL `https://aclanthology.org/2023.findings-emnlp.68`.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenhang Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, K. Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Yu Bowen, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xing Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *ArXiv*, abs/2309.16609, 2023. URL `https://api.semanticscholar.org/CorpusID:263134555`.

Bernd Bohnet, Vinh Q Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, et al. Attributed question answering: Evaluation and modeling for attributed large language models. *arXiv preprint arXiv:2212.08037*, 2022.

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Roi Cohen, Mor Geva, Jonathan Berant, and Amir Globerson. Crawling the internal knowledge-base of language models. In Andreas Vlachos and Isabelle Augenstein, editors, *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1856–1869, Dubrovnik, Croatia, May 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-eacl.139. URL `https://aclanthology.org/2023.findings-eacl.139`.

Roi Cohen, May Hamri, Mor Geva, and Amir Globerson. LM vs LM: Detecting factual errors via cross examination. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12621–12640, Singapore, December 2023b. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.778. URL https://aclanthology.org/2023.emnlp-main.778.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024a.

Roi Cohen, Konstantin Dobler, Eden Biran, and Gerard de Melo. I Don't Know: Explicit modeling of uncertainty with an [IDK] token. *Advances in Neural Information Processing Systems*, 37: 10935–10958, 2024b.

Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.21. URL https://aclanthology.org/2020.emnlp-main.21.

Ashwin Devaraj, William Sheffield, Byron Wallace, and Junyi Jessy Li. Evaluating factuality in text simplification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7331–7345, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.506. URL https://aclanthology.org/2022.acl-long.506.

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

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031, 2021. doi: 10.1162/tacl_a_00410. URL https://aclanthology.org/2021.tacl-1.60.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361, 2021a. doi: 10.1162/tacl_a_00370. URL https://aclanthology.org/2021.tacl-1.21.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.751. URL https://aclanthology.org/2023.emnlp-main.751/.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/guo17a.html.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *arXiv preprint arXiv:2305.01610*, 2023.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), mar 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023a.

Albert Qiaochu Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L'elio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b. *ArXiv*, abs/2310.06825, 2023b. URL `https://api.semanticscholar.org/CorpusID:263830494`.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977, 2021. doi: 10.1162/tacl_a_00407. URL `https://aclanthology.org/2021.tacl-1.57`.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zachary Dodds, Nova Dassarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, John Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom B. Brown, Jack Clark, Nicholas Joseph, Benjamin Mann, Sam McCandlish, Christopher Olah, and Jared Kaplan. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Sanyam Kapoor, Nate Gruver, Manley Roberts, Katherine Collins, Arka Pal, Umang Bhatt, Adrian Weller, Samuel Dooley, Micah Goldblum, and Andrew Gordon Wilson. Large language models must be taught to know what they don't know. *arXiv preprint arXiv:2406.08391*, 2024.

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, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL `https://aclanthology.org/Q19-1026`.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. Bloom: A 176b-parameter open-access multilingual language model. 2023.

Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

Stephanie C. Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Annual Meeting of the Association for Computational Linguistics*, 2021. URL `https://api.semanticscholar.org/CorpusID:237532606`.

Hongyuan Lu, Wai Lam, Hong Cheng, and Helen Meng. On controlling fallback responses for grounded dialogue generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2591–2601, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.204. URL `https://aclanthology.org/2022.findings-acl.204`.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 2022.

Laura Manduchi, Kushagra Pandey, Robert Bamler, Ryan Cotterell, Sina Däubener, Sophie Fellenz, Asja Fischer, Thomas Gärtner, Matthias Kirchler, Marius Kloft, Yingzhen Li, Christoph Lippert, Gerard de Melo, Eric Nalisnick, Björn Ommer, Rajesh Ranganath, Maja Rudolph, Karen Ullrich, Guy Van den Broeck, Julia E Vogt, Yixin Wang, Florian Wenzel, Frank Wood, Stephan Mandt, and Vincent Fortuin. On the challenges and opportunities in Generative AI. *arXiv preprint arXiv:2403.00025*, 2024. URL `https://arxiv.org/abs/2403.00025`.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL `https://aclanthology.org/2020.acl-main.173`.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=-h6WAS6eE4`.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing English math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.92. URL `https://aclanthology.org/2020.acl-main.92`.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. *arXiv preprint arXiv:2305.15852*, 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.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

Jeff Z. Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, Russa Biswas, Gerard de Melo, Angela Bonifati, Edlira Vakaj, Mauro Dragoni, and Damien Graux. Large Language Models and Knowledge Graphs: Opportunities and Challenges. *Transactions on Graph Data and Knowledge*, 1(1):2:1–2:38, 2023. doi: 10.4230/TGDK.1.1.2. URL `https://drops.dagstuhl.de/entities/document/10.4230/TGDK.1.1.2`.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.168. URL `https://aclanthology.org/2021.naacl-main.168`.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

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

Hannah Rashkin, Vitaly Nikolaev, Matthew Lamm, Lora Aroyo, Michael Collins, Dipanjan Das, Slav Petrov, Gaurav Singh Tomar, Iulia Turc, and David Reitter. Measuring attribution in natural language generation models. *Computational Linguistics*, 49(4):777–840, 2023.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*, 2020.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL `https://aclanthology.org/N19-1421`.

Derek Tam, Anisha Mascarenhas, Shiyue Zhang, Sarah Kwan, Mohit Bansal, and Colin Raffel. Evaluating the factual consistency of large language models through news summarization. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5220–5255, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.322. URL `https://aclanthology.org/2023.findings-acl.322`.

Yi Tay, Vinh Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Jiawen Wu, Xipeng Qiu, and Xuanjing Huang. Do large language models know what they don't know? *arXiv preprint arXiv:2305.18153*, 2023.

Gal Yona, Roee Aharoni, and Mor Geva. Narrowing the knowledge evaluation gap: Open-domain question answering with multi-granularity answers. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6737–6751, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.365. URL `https://aclanthology.org/2024.acl-long.365/`.

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. Answering questions by meta-reasoning over multiple chains of thought. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5942–5966, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.364. URL `https://aclanthology.org/2023.emnlp-main.364`.

Lei Yu, Meng Cao, Jackie Chi Kit Cheung, and Yue Dong. Mechanistic understanding and mitigation of language model non-factual hallucinations. *arXiv preprint arXiv:2403.18167*, 2024.

Qinan Yu, Jack Merullo, and Ellie Pavlick. Characterizing mechanisms for factual recall in language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9924–9959, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.615. URL `https://aclanthology.org/2023.emnlp-main.615/`.

Xiang Yue, Boshi Wang, Ziru Chen, Kai Zhang, Yu Su, and Huan Sun. Automatic evaluation of attribution by large language models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 4615–4635, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.307. URL `https://aclanthology.org/2023.findings-emnlp.307`.

Hanning Zhang, Shizhe Diao, Yong Lin, Yi Fung, Qing Lian, Xingyao Wang, Yangyi Chen, Heng Ji, and Tong Zhang. R-tuning: Instructing large language models to say 'I don't know'. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7113–7139, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.394. URL `https://aclanthology.org/2024.naacl-long.394/`.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*, 2023.

Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Lei Shen, Zihan Wang, Andi Wang, Yang Li, et al. Codegeex: A pre-trained model for code generation with multilingual benchmarking on humaneval-x. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5673–5684, 2023.

Victor Zhong, Weijia Shi, Wen tau Yih, and Luke Zettlemoyer. RoMQA: A benchmark for robust, multi-evidence, multi-answer question answering. In *Conference on Empirical Methods in Natural Language Processing*, 2022. URL `https://api.semanticscholar.org/CorpusID:253116788`.

A Limitations

While our analysis provides compelling evidence for the existence of linearly accessible uncertainty representations in LLMs, it is limited to linear probes and does not explore more complex, nonlinear structures that may further explain model behavior. Our evaluation focuses on a fixed set of models and datasets, which, although diverse, may not capture the full variability seen in real-world applications or domain-specific tasks. Additionally, correctness is treated as a proxy for uncertainty, which may not fully align with how uncertainty manifests in open-ended or ambiguous generation scenarios. Finally, the performance of our classifiers may also be influenced by dataset-specific biases, potentially limiting generalizability.

B Computer Resources

In our experiments we use one NVIDIA A100 80G GPU.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: We run extensive experiments to support our claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: See Appendix A.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [NA]

   Justification: N/A

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: See section 2 and 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

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

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

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

   Answer: [No]

   Justification: Will be released with the camera-ready.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: See section 3.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Our experiments have been conducted at large scale including 9 different models and 16 different datasets, means that it's very likely that statistical errors are negligible in this case.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: See Appendix B

   Guidelines:

   - The answer NA means that the paper does not include experiments.

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We reviewed the guidelines and are in conformance.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: See Appendix A.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [No]

    Justification: We do not include safeguards.

Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We cite all used models, datasets and methods.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Full details of all assets will be made available.

    Guidelines:

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

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

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

Answer: [NA]

Justification: N/A

Guidelines:

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

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

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

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: See section 3.

    Guidelines:

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