POSIX: A Prompt Sensitivity Index For Large Language Models

Anwoy Chatterjee[*†]
Dept. of Electrical Engineering
Indian Institute of Technology Delhi
anwoy.chatterjee@ee.iitd.ac.in

H S V N S Kowndinya Renduchintala[†]
Media and Data Science Research
Adobe Inc., India
rharisrikowndinya333@gmail.com

Sumit Bhatia
Media and Data Science Research
Adobe Inc., India
sumit.bhatia@adobe.com

Tanmoy Chakraborty
Dept. of Electrical Engineering
Indian Institute of Technology Delhi
tanchak@iitd.ac.in

Abstract

Despite their remarkable capabilities, Large Language Models (LLMs) are found to be surprisingly sensitive to minor variations in prompts, often generating significantly divergent outputs in response to minor variations in the prompts, such as spelling errors, alteration of wording or the prompt template. However, while assessing the quality of an LLM, the focus often tends to be solely on its performance on downstream tasks, while very little to no attention is paid to prompt sensitivity. To fill this gap, we propose POSIX – a novel ***PrOmpt Sensitivity IndeX*** as a reliable measure of prompt sensitivity, thereby offering a more comprehensive evaluation of LLM performance. The key idea behind POSIX is to capture the relative change in log-likelihood of a given response upon replacing the corresponding prompt with a different intent-preserving prompt. We provide thorough empirical evidence demonstrating the efficacy of POSIX in capturing prompt sensitivity and subsequently use it to measure and thereby compare prompt sensitivity of various open-source LLMs. We find that merely increasing the parameter count or instruction tuning does not necessarily reduce prompt sensitivity whereas adding some few-shot exemplars, even just one, almost always leads to significant decrease in prompt sensitivity. We also find that alterations to prompt template lead to the highest sensitivity in the case of MCQ-type tasks, whereas paraphrasing results in the highest sensitivity in open-ended generation tasks. The code for reproducing our results is open-sourced at https://github.com/kowndinya-renduchintala/POSIX.

1 Introduction

Large Language Models (LLMs) are pre-trained on enormous amounts of text data using the *next-token-prediction* objective and they can perform a variety of NLP tasks via *"prompting"* (Brown et al., 2020; Kojima et al., 2022; Almazrouei et al., 2023; Liu et al., 2023; Touvron et al., 2023). However, LLMs have been found to be surprisingly sensitive even to the smallest of variations in prompts that do not significantly alter its meaning – such as wording, prompt template or even minor spelling errors – so much so that *prompt engineering*, which is a process of iteratively tuning prompts to elicit desired responses, has become a widespread practice (Reynolds and McDonell, 2021).

Despite prompt sensitivity being a crucial aspect for assessing the usability of an LLM, standard evaluation benchmarks such as MMLU (Hendrycks et al., 2021) or BBH (Suzgun et al., 2022) focus predominantly on performance metrics like exact match, leaving prompt sensitivity sidelined. Similarly, the model cards and blog posts announcing the most commonly used LLMs often do not contain prompt sensitivity analysis at all (AI@Meta, 2024). However, from a user-centric perspective, models with low prompt sensitivity are generally preferred over highly prompt-sensitive ones, even if both perform similarly on standard benchmarks. This is largely because a real-world user may not always be able to formulate the "optimal" prompt everytime. Moreover, a universally applicable optimal prompt that can work *across* different model architectures may not even exist. Therefore, it becomes essential to develop a dedicated procedure to systematically evaluate and quantify the sensitivity of LLMs towards *intent-preserving* (or, *intent-aligned*) variations in prompts.

While the exploration of the topic of quantifying prompt sensitivity is limited, there exist a few works which study prompt sensitivity and attempt to quantify it. For instance, the HELM benchmark (Liang et al., 2023) implements various kinds of perturbations to the prompts, such as typos and misspellings, and reports the exact match score of

[†]These two authors contributed **equally** to this work.
[*]Work done during internship at Media and Data Science Research (MDSR) Lab, Adobe Inc.

Findings of the Association for Computational Linguistics: EACL 2024, pages 14550–14565

responses obtained using the perturbed prompts. Sclar et al. (2023) studied the sensitivity to variations in prompt templates. They introduced the concept of *performance spread* – defined as the difference between the highest and lowest average performances observed across a wide range of prompt templates – and used this as a surrogate for prompt sensitivity. Likewise, Lu et al. (2023) first generated responses using variations of the same prompt and then used variation-ratio (Freeman, 1965) as the surrogate for sensitivity.

Despite being in the right direction, the existing efforts to quantify prompt sensitivity lack nuance. Relying on variations in model accuracy as a proxy for sensitivity ignores the model's behavior in case of incorrect responses, failing to distinguish between a model that consistently generates the same incorrect response every time (*low sensitivity*) and one that generates a different incorrect response with each prompt variation (*high sensitivity*). Additionally, response distribution matters: a model producing a consistent response in all but one case is less sensitive than one generating two different responses - each equally frequent. The exact match score is also brittle, heavily penalizing minor wording differences in correct responses. Furthermore, models might show significant variance in the log-likelihoods of responses for different prompt variations even when the responses are identical. Finally, the existing studies are also limited to deterministic responses, like multiple-choice questions or short answers (Kwiatkowski et al., 2019) and do not consider the sensitivity in the case of open-ended generative tasks.

With this context, in this work, we focus on the following research question: *given a prompt along with its intent-preserving variations and the corresponding set of responses generated by a language model, how do we measure the sensitivity of the LLM on the given set of prompts such that the measure incorporates the following four key factors?*

- ***Response Diversity:*** *A higher number of unique responses generated for a given set of intent-preserving prompts should indicate higher sensitivity.*
- ***Response Distribution Entropy:*** *Higher entropy of the distribution of response frequencies (how often each unique response appears) should indicate higher sensitivity.*
- ***Semantic Coherence:*** *Lower semantic similarity among generated responses should contribute to higher sensitivity.*
- ***Variance in Confidence:*** *Higher variance in the log-likelihood of the same response should contribute to higher sensitivity.*

To the best of our knowledge, ours is the first attempt towards developing a prompt sensitivity index that takes all of the above four factors into account and that works across different kinds of prompt variations and different kinds of tasks, including open-ended generation with arbitrary response lengths.

As a first step towards addressing the research question, we identify the key property that should hold for an *ideal LLM* that is not sensitive to variations in prompts, provided the underlying intent is unchanged. For this LLM, the probability of generating a certain response should remain almost the same even if minor changes are introduced in the prompt. For instance, let x_1 and x_2 be two prompts such that one is an intent-preserving variant of the other and let y_1 and y_2 be the corresponding responses generated by the LLM under consideration. Then the assumption, which turns out to be the cornerstone of our index , is that for an LLM that is not sensitive, $\mathbb{P}(y_1|x_1)$ should not change much if x_1 is replaced by its intent-preserving variant, x_2, i.e., $\mathbb{P}(y_1|x_1) \approx \mathbb{P}(y_1|x_2)$. Similarly the other way round — $\mathbb{P}(y_2|x_2) \approx \mathbb{P}(y_2|x_1)$.

We emphasize that this notion of sensitivity does not consider the ground-truth response for quantifying sensitivity; instead, it compares the likelihood of generating various responses with different intent-preserving variants of the same prompt. In that sense, it is completely orthogonal to metrics like exact match (which are purely performance-based) and adds a new dimension altogether for evaluating LLMs.

The main contributions of our work can be summarized as follows:

- We introduce `POSIX` – a novel prompt sensitivity index that can act as a reliable measure of sensitivity of LLMs towards intent-preserving variations of prompts (Section 3). We empirically show in Section 5.1 that `POSIX` incorporates all the four factors listed above.
- We use `POSIX` to compute prompt sensitivity of different models and variation types, and show the effect of various aspects on the prompt sensitivity of LLMs, e.g., that increase in parameter count or instruction tuning do not necessarily decrease prompt sensitivity, and incorporating few-shot exemplars, even just a single exemplar, makes a huge difference in

reducing prompt sensitivity (Section 5).

- We reveal an interesting observation based on POSIX computation — variations in template bring about maximum sensitivity in the case of MCQ-type tasks while variations in wording lead to maximum effect in the case of open-ended generation tasks (Section 5).

2 Related Work

Sensitivity of LLMs to Prompt Variations. The in-context learning ability of LLMs (Brown et al., 2020) makes them highly versatile, enabling them to perform a wide range of tasks through *prompting*, often without the need for further fine-tuning (Radford et al., 2019; Raffel et al., 2020; Gao et al., 2021). However, the robustness of in-context learning is often questioned (Weber et al., 2023), with many studies showing that the output from LLMs is heavily dependent on aspects like the selection and ordering of in-context examples (Liu et al., 2022; Su et al., 2022; Lu et al., 2022; Zhao et al., 2021), choice of input labels (Min et al., 2022), or the phrasing of instruction given in the prompt (Gu et al., 2023; Sun et al., 2024). Apart from these aspects, LLMs are also observed to be highly sensitive to slight modifications in the structure or wordings of the prompts, even though their semantic meaning remains the same. Many prior works (Arora et al., 2022; Leidinger et al., 2023; Sclar et al., 2023; Voronov et al., 2024; Mizrahi et al., 2024) have studied this issue of sensitivity of LLMs to minor alterations in the input prompt.

Few of these works (Leidinger et al., 2023; Mizrahi et al., 2024; Voronov et al., 2024) have also called for extending the evaluation benchmarks – they argue that instead of evaluating on a single instance of a prompt, the benchmarks should include multiple variants for each prompt to account for the divergence in behaviour of the models to prompt variations. While for most existing benchmarks like MMLU (Hendrycks et al., 2021) or BIG-bench (Srivastava et al., 2022; Suzgun et al., 2022), the performance is reported for a single template of the prompts, the LMentry (Efrat et al., 2023) benchmark uses three templates for each task in it and reports the average performance over them. Also, recently, Polo et al. (2024) proposed *PromptEval*, a method to facilitate efficient evaluation of LLMs on any benchmark with multiple prompt templates under a limited budget. Zhu et al. (2023) introduced *Promptbench* for evaluating the robustness of LLMs to variations done in prompts with adversarial intent – they observed that almost all LLMs lack robustness towards adversarial prompts. They quantified robustness using *Performance Drop Rate*, which measures the relative drop in performance when perturbations are introduced into the prompt.

In this work, we also advocate for benchmarks with multiple variations of the same prompt. However, instead of relying on measures based on performance alone for measuring sensitivity or robustness, we argue the need for a comprehensive measure that can capture the prompt sensitivity of LLMs effectively.

Prompt Engineering. Due to such extensive variation in the performance of LLMs on slight modifications in input prompt, it is crucial to query LLMs with the optimal prompt to get the desired output. Prompt engineering is the practice of crafting tailored prompts for input to the LLMs to guide them towards the intended responses. Though for real-world use cases, users often perform prompt engineering manually, prior studies have also proposed ways to automate this process (Deng et al., 2022). Another method for obtaining a better input prompt is *Meta-prompting* (Reynolds and McDonell, 2021; Zhou et al., 2023; Ye et al., 2024), which aims to improve a prompt iteratively with the help of an LLM itself through further prompting. Furthermore, for tasks involving reasoning, *Chain-of-Thought* prompting (Wei et al., 2023) has been observed to be very effective.

For designing a few-shot prompt, the choice of in-context examples plays a crucial role in the performance of LLMs. While existing studies (Liu et al., 2022; Min et al., 2022) show that examples which are semantically similar to the input work the best, in some cases selecting diverse examples can be beneficial (Su et al., 2022; Min et al., 2022). To maximize the potential of LLMs, it is therefore crucial to be aware of the best practices for prompting.

3 POSIX: Prompt Sensitivity Index

Based on the notion of sensitivity introduced in Section 1, we will first define which intent-preserving variations we consider in this work and describe POSIX in detail, including a brief description of the design choices involved.

3.1 Preliminaries

Definition 3.1 *Any two prompts x_1 and x_2 are said to be **intent-aligned** despite variations in their wording or template or inclusion of minor spelling errors, if they are designed to elicit responses from a language model based on the same underlying goal, intent or meaning.*

Definition 3.2 *A set of prompts $\mathbf{X} = \{x_i\}_{i=1}^N$ is said to be an **intent-aligned prompt set** if and only if for all $1 \leq i \neq j \leq N$, $x_i \in \mathbf{X}$ and $x_j \in \mathbf{X}$ are intent-aligned.*

3.2 Defining the Prompt Sensitivity Index

Definition 3.3 *Let $\mathbf{X} = \{x_i\}_{i=1}^N$ be an intent-aligned prompt set and $\mathbf{Y} = \{y_i\}_{i=1}^N$ be the set of corresponding responses generated by a language model $\mathcal{M}$, i.e., y_i is the response generated by $\mathcal{M}$ when prompted using x_i. The sensitivity of the model $\mathcal{M}$ on $\mathbf{X}$ is defined as*

$$\psi_{\mathcal{M},\mathbf{X}} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{1}{L_{y_j}} \left| \log \frac{\mathbb{P}_{\mathcal{M}}(y_j|x_i)}{\mathbb{P}_{\mathcal{M}}(y_j|x_j)} \right|$$

*where N is the cardinality of $\mathbf{X}$, L_{y_j} is the number of tokens in y_j, $\mathbb{P}_{\mathcal{M}}(y_j|x_i)$ is the probability of the model $\mathcal{M}$ generating the response y_j **given** the prompt x_i, and $\mathbb{P}_{\mathcal{M}}(y_j|x_j)$ is the probability of the model $\mathcal{M}$ generating the response y_j **given** the prompt x_j.*

Definition 3.4 *(POSIX) Given a language model $\mathcal{M}$ and a dataset $\mathcal{D} = \{\mathbf{X}_i\}_{i=1}^M$ of M intent-aligned prompt sets (X_i's), the prompt sensitivity index (POSIX) for the language model $\mathcal{M}$ on the dataset $\mathcal{D}$ is defined as*

$$\texttt{POSIX}_{\mathcal{D},\mathcal{M}} = \frac{1}{M} \sum_{i=1}^{M} \psi_{\mathcal{M},\mathbf{X}_i}$$

3.3 What does $\psi_{\mathcal{M},\mathbf{X}}$ Capture?

As mentioned briefly in Section 1, if we have two intent-aligned prompts x_i and x_j and the corresponding responses y_i and y_j generated by an LLM $\mathcal{M}$, we would essentially like to capture how different are $\mathbb{P}_{\mathcal{M}}(y_i|x_i)$ and $\mathbb{P}_{\mathcal{M}}(y_i|x_j)$ (and also similarly for $\mathbb{P}_{\mathcal{M}}(y_j|x_j)$ and $\mathbb{P}_{\mathcal{M}}(y_j|x_i)$). In order to make the sensitivity measure comparable across different intent-aligned prompt sets as well as across models, we need to remove the dependence on the overall scale of the model's probability distributions. Therefore, we consider the ratios $\frac{\mathbb{P}_{\mathcal{M}}(y_j|x_i)}{\mathbb{P}_{\mathcal{M}}(y_j|x_j)}$ which are immune to scale. And, since we only need to look at relative change in probabilities by replacing x_i with x_j or vice-versa, we convert the probability ratio to the logarithmic scale and also use the absolute value on top of it. Furthermore, in order to accommodate for arbitrary response lengths, we use length normalization for each term in the summation.

We now look at conceptual explanations for why $\psi_{\mathcal{M},\mathbf{X}}$ incorporates four properties listed in Section 1 while deferring the empirical evidence of the same to Section 5.1.

3.3.1 POSIX and Response Diversity

While not explicitly evident in the expression of $\psi_{\mathcal{M},\mathbf{X}}$, response diversity contributes indirectly to higher $\psi_{\mathcal{M},\mathbf{X}}$. Say if two responses, y_i and y_j, are significantly different, then their log-likelihoods (given a prompt) are likely to be significantly different i.e., the terms in summation, $\left| \log \frac{\mathbb{P}_{\mathcal{M}}(y_j|x_i)}{\mathbb{P}_{\mathcal{M}}(y_j|x_j)} \right|$, become large, thereby leading to greater $\psi_{\mathcal{M},\mathbf{X}}$ overall.

3.3.2 POSIX and Response Distribution Entropy

By response distribution entropy, we mean the entropy of the distribution of response frequencies, i.e., how many times each unique response appears in $\mathbf{Y}$. A higher entropy indicates the tendency of the model to generate divergent responses more often – consequently, the magnitude of the log-likelihood ratios in the summation of Definition 3.3 will tend to be high, resulting in an uptick in the value of $\psi_{\mathcal{M},\mathbf{X}}$ with increase in response distribution entropy.

3.3.3 POSIX and Semantic Coherence

When the responses to intent-aligned prompts are semantically similar, i.e., the average cosine similarity between their embeddings is high, then intuitively the model is less sensitive. This is also captured by $\psi_{\mathcal{M},\mathbf{X}}$ because if x_i and x_j both generate semantically similar responses y_i and y_j, then the probability of generating y_j typically does not differ significantly for the two intent-aligned prompts x_i and x_j and so in such cases, the individual terms in the summation are low, leading to a lower $\psi_{\mathcal{M},\mathbf{X}}$ overall.

Remark: It may so happen that the responses to two intent-aligned prompts x_i and x_j are semantically equivalent but involve very different word choices and still $\mathbb{P}_{\mathcal{M}}(y_j|x_j)$ and $\mathbb{P}_{\mathcal{M}}(y_j|x_i)$ can

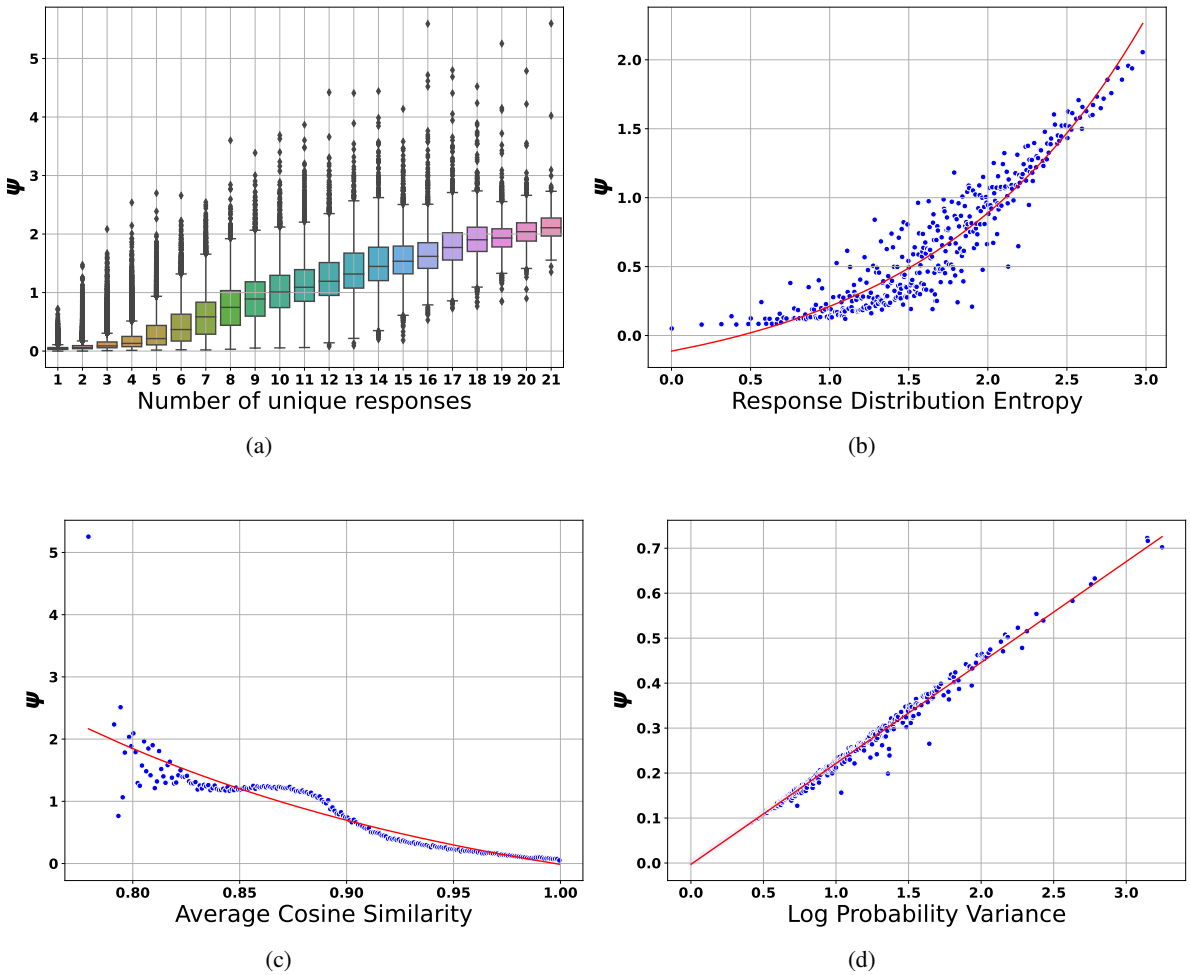


Figure 1: Correlation plots of ψ with each of the four factors described in Section 3.3 in the case of MMLU: (a) Response Diversity; (b) Response Distribution Entropy; (c) Semantic Coherence; (d) Variance in Confidence.

be significantly different. In fact, depending on the model and input prompts, this can happen even if y_i and y_j are the exact same strings! Therefore, in Section 3.3.4, we also consider the variance in probabilities even if the strings are exactly the same. This is also a reason why average cosine similarity between the generated responses cannot by itself be employed as a prompt sensitivity metric.

3.3.4 POSIX and Variance in Confidence

Consider the case where all responses to the prompts in $\mathbf{X}$ are exactly the same, i.e., all y_j's are the same. In such a case, should we call the model to be not sensitive at all? Upon a closer look, we can realize that the model would still be considered as sensitive if there is a notable variation in the likelihood of generating the response with a change in the input prompt. As evident from Definition 3.3, our proposed index $\psi_{\mathcal{M},\mathbf{X}}$ directly measures how divergent the log-likelihoods are for different intent-aligned prompts, thereby capturing the subtle nuances in response generation which contribute towards the sensitivity of the models.

4 Experimental Setup

To analyse the effectiveness of POSIX in capturing various facets of sensitivity as described in Section 3 and to quantify and compare the prompt sensitivity of various LLMs using it, we experiment on Massive Multitask Language Understanding benchmark, or MMLU (Hendrycks et al., 2021), for classification tasks (posed as MCQ questions), and on Alpaca (Taori et al., 2023) for open-ended generation tasks. We also include Big Bench Hard, or BBH (Suzgun et al., 2022) as an additional dataset in Appendix E. MMLU contains about $14,000$ prompt-response pairs from 57 different domains. For open-ended generation task, we sample $5,000$ questions from Alpaca.

We consider a total of eight LLMs from three

Model	MMLU-ZeroShot				Alpaca-ZeroShot			
	Spelling Errors	Prompt Templates	Paraphrases	Mixture	Spelling Errors	Prompt Templates	Paraphrases	Mixture
Llama-2-7b	$0.083_{\pm 0.073}$	$1.12_{\pm 0.377}$	$0.160_{\pm 0.160}$	$0.821_{\pm 0.272}$	$0.146_{\pm 0.115}$	$0.202_{\pm 0.103}$	$0.252_{\pm 0.192}$	$0.271_{\pm 0.158}$
Llama-2-7b-chat	$0.082_{\pm 0.103}$	$0.809_{\pm 0.283}$	$0.135_{\pm 0.189}$	$0.444_{\pm 0.258}$	$0.246_{\pm 0.175}$	$0.164_{\pm 0.139}$	$0.66_{\pm 0.33}$	$0.500_{\pm 0.229}$
Llama-3-8b	$0.086_{\pm 0.097}$	$1.106_{\pm 0.612}$	$0.11_{\pm 0.109}$	$0.641_{\pm 0.383}$	$0.123_{\pm 0.091}$	$0.150_{\pm 0.107}$	$0.249_{\pm 0.175}$	$0.239_{\pm 0.136}$
Llama-3-8b-chat	$0.087_{\pm 0.09}$	$1.048_{\pm 0.612}$	$0.134_{\pm 0.126}$	$0.650_{\pm 0.421}$	$0.184_{\pm 0.152}$	$0.15_{\pm 0.13}$	$0.413_{\pm 0.259}$	$0.357_{\pm 0.201}$
Mistral-7B	$0.065_{\pm 0.06}$	$1.222_{\pm 0.571}$	$0.108_{\pm 0.114}$	$0.672_{\pm 0.303}$	$0.18_{\pm 0.14}$	$0.217_{\pm 0.148}$	$0.242_{\pm 0.181}$	$0.295_{\pm 0.181}$
Mistral-7B-Instruct	$0.105_{\pm 0.098}$	$1.464_{\pm 0.528}$	$0.126_{\pm 0.112}$	$0.886_{\pm 0.328}$	$0.195_{\pm 0.130}$	$0.124_{\pm 0.069}$	$0.296_{\pm 0.236}$	$0.272_{\pm 0.152}$
OLMo-7B-Base	$0.197_{\pm 0.207}$	$1.672_{\pm 0.383}$	$0.189_{\pm 0.164}$	$1.134_{\pm 0.286}$	$0.355_{\pm 0.305}$	$0.369_{\pm 0.095}$	$0.281_{\pm 0.199}$	$0.448_{\pm 0.227}$
OLMo-7B-Instruct	$0.527_{\pm 0.485}$	$1.499_{\pm 0.384}$	$0.831_{\pm 0.595}$	$1.413_{\pm 0.474}$	$0.646_{\pm 0.378}$	$0.192_{\pm 0.113}$	$0.633_{\pm 0.382}$	$0.62_{\pm 0.312}$

Table 1: POSIX computed for 8 different models and 4 different prompt variation types on both MCQs (MMLU) and open-ended generation (Alpaca). The mixture variant consists of equal proportion of the other three variations.

families: LLaMA, Mistral, and OLMo. These models include LLaMA-2 7B (base and chat variants) (Touvron et al., 2023), LLaMA-3 8B (base and instruct variants), Mistral 7B (base and instruct variants) (Jiang et al., 2023) and OLMo 7B (base and instruct variants) (Groeneveld et al., 2024). All our experiments were run on 8 NVIDIA A100-SXM4-80GB GPUs.

For each prompt, we generate 60 variants such that they are *intent-aligned* with each other and also with the original prompt. These 60 variants are composed of equal proportion of three types of variations — introduction of minor spelling errors, alteration of the prompt template and paraphrasing/re-wording of the prompt. To study the effect of variation type on sensitivity, each prompt variant is generated using one of these three possible alterations. The variation types are discussed in detail below:

Spelling Errors: To introduce spelling errors, we randomly select one, two, four or eight tokens from the question in the prompt and introduce one of four possible spelling errors: (i) *Insertion*, in which a random letter is added within the token; (ii) *Omission*, in which a letter at a randomly chosen position is deleted; (iii) *Transposition*, in which two adjacent letters are swapped; or (iv) *Substitution*, in which a letter at a randomly chosen position is replaced with one of its adjacent letters on the keyboard. The specific error to be applied to each token is chosen randomly.

The 20 variations for spelling errors are derived from the combinations of the number of tokens with errors (one, two, four, eight) and five different seeds to control randomness. These types of spelling errors are based on the study by Brooks et al. (1993).

Prompt Templates: For altering the prompt template, we design 20 different templates based on the grammar defined by Sclar et al. (2023). These templates are designed to maintain the core meaning of the prompt while altering its structure. The prompt templates used in our study are listed in Appendix B.

Paraphrases: To create paraphrased variations of the prompts, we use GPT-3.5-Turbo to generate 20 paraphrases for each original prompt (The full dataset of generated paraphrases is open-sourced however a few examples of generated paraphrases are also present in Appendix F for reference). These paraphrases are such that they rephrase the question while preserving its original intent and meaning.

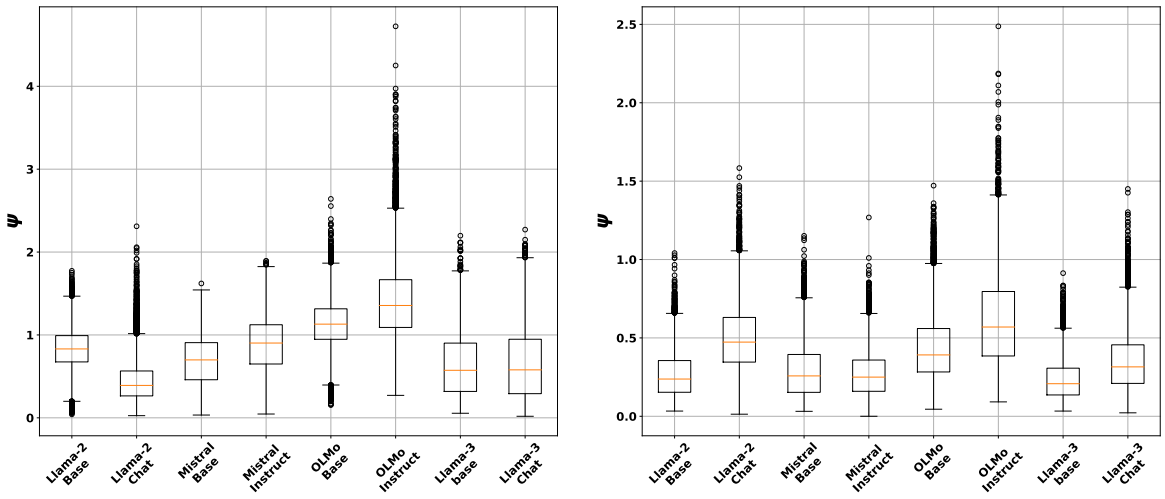
For each prompt, we compute POSIX for each type of variation, based on Definition 3.3, using an intent-aligned prompt set that includes the original prompt and its 20 corresponding variants. Additionally, for the **Mixture** category, we calculate POSIX using a set of 21 prompt variants sampled uniformly from the three variation categories.

For experiments on MMLU, we generate five tokens as output, as the tasks are of multiple-choice question-answering format; whereas for the open-ended questions in Alpaca, 30 tokens are generated. POSIX is, however, comparable for arbitrary response lengths as it is normalized by the number of tokens in the response.

5 Results and Analysis

5.1 Evaluating the Efficacy of POSIX

We now empirically investigate if POSIX incorporates the four factors described in Section 3 by looking at correlation plots of POSIX with those factors. For the plots, we combine data from all types of prompt variations and all models listed in Section 4. Additionally, for computing the cosine similarity between generated responses, we use an off-the-



(a) MMLU (MCQs)

(b) Alpaca (Open-ended generation)

Figure 2: Box plots depicting the distribution of $\psi_{\mathcal{M},\mathbf{X}}$ for different instances of $\mathcal{M}$. The first plot corresponds to $\mathbf{X}$'s from MMLU dataset (MCQs) and the second plot corresponds to $\mathbf{X}$'s from the Alpaca dataset (open-ended generation).

shelf Sentence Transformer model (GTE-large). Figure 1 shows the correlation plots between POSIX and each of the four properties described in Section 3. For response distribution entropy, average cosine similarity and the log-probability variance, to observe the trend, we first bin the x-axes and corresponding average ψ of the bins are plotted. The number of unique responses, the response distribution entropy and the log-probability variance (in case all responses are identical) — all have positive correlation with POSIX and the average cosine similarity between the responses is negatively correlated with POSIX. Thus, the plots serve as an empirical validation for the fact that POSIX incorporates each of the factors described in Section 3. Please refer to Appendix G for the correlation plots of open-ended generation (Alpaca).

5.2 Effect of Instruction Tuning on Sensitivity

Table 1 presents the POSIX values of various models obtained for different variation types on MMLU and Alpaca (Please refer to Appendix E for results on BBH). We observe that chat or instruct versions of the models are generally less sensitive than the corresponding base models in the case of template variations on MMLU, with Mistral being the only exception. However, in the other categories of variations, the instruct versions tend to be more sensitive than their base models. Especially for open-ended generation tasks, i.e., on Alpaca, the higher sensitivity of instruct versions is even more pronounced. Note that this implies that instruction tuning does not necessarily improve model sensitivity. Figure 2 shows the distribution of POSIX values of all models for the *Mixture* of variation types (please refer to Appendix H for the other variation types). We clearly observe the instruct models to have higher sensitivity than the base ones, more so on Alpaca. Furthermore, the Mistral models seem to have the least divergence in sensitivity between the base and instruct variants whereas the OLMo models have the most disparity.

As the base models undergo both instruction tuning and alignment on human preferences to obtain the instruct versions, the above observations are a cumulative effect of instruction tuning and alignment procedures. To disentangle their consequences and study the effect of only instruction tuning on sensitivity, we separately fine-tune LLaMA-2 7B and Mistral 7B base models on the entire FLAN dataset (Wei et al., 2022). We call these fine-tuned models Llama-2-7b-FLAN and Mistral-7B-FLAN, respectively. Their POSIX values are reported in Table 3. In most cases, the chat version is better than FLAN-only models in terms of sensitivity, except in case of prompt template for MMLU, where FLAN-only models significantly outperform the chat versions. We hypothesize that this is due to the nature of the FLAN dataset, which contains a huge focus on MCQs and various prompt

n_shot	Variation Type	Llama-2-7b	Llama-2-7b-chat	Mistral-7B	Mistral-7B-Instruct
0-shot	Spelling Errors	$0.083_{\pm 0.073}$	$0.082_{\pm 0.103}$	$0.065_{\pm 0.06}$	$0.105_{\pm 0.098}$
	Prompt Templates	$1.12_{\pm 0.377}$	$0.809_{\pm 0.283}$	$1.222_{\pm 0.571}$	$1.464_{\pm 0.0.528}$
	Paraphrases	$0.16_{\pm 0.16}$	$0.135_{\pm 0.189}$	$0.108_{\pm 0.115}$	$0.126_{\pm 0.112}$
1-shot	Spelling Errors	$0.026_{\pm 0.021}$	$0.048_{\pm 0.066}$	$0.042_{\pm 0.039}$	$0.087_{\pm 0.065}$
	Prompt Templates	$0.513_{\pm 0.347}$	$0.357_{\pm 0.169}$	$0.2_{\pm 0.244}$	$1.387_{\pm 0.707}$
	Paraphrases	$0.035_{\pm 0.031}$	$0.064_{\pm 0.0.07}$	$0.046_{\pm 0.045}$	$0.085_{\pm 0.081}$
2-shot	Spelling Errors	$0.027_{\pm 0.024}$	$0.049_{\pm 0.07}$	$0.042_{\pm 0.041}$	$0.085_{\pm 0.072}$
	Prompt Templates	$0.482_{\pm 0.38}$	$0.272_{\pm 0.117}$	$0.225_{\pm 0.247}$	$1.128_{\pm 0.773}$
	Paraphrases	$0.036_{\pm 0.035}$	$0.065_{\pm 0.074}$	$0.047_{\pm 0.047}$	$0.085_{\pm 0.09}$
3-shot	Spelling Errors	$0.028_{\pm 0.024}$	$0.051_{\pm 0.073}$	$0.043_{\pm 0.041}$	$0.088_{\pm 0.073}$
	Prompt Templates	$0.554_{\pm 0.433}$	$0.249_{\pm 0.091}$	$0.23_{\pm 0.247}$	$1.101_{\pm 0.775}$
	Paraphrases	$0.039_{\pm 0.039}$	$0.068_{\pm 0.077}$	$0.047_{\pm 0.047}$	$0.086_{\pm 0.0.98}$

Table 2: POSIX computed for Llama-2 and Mistral models on the MMLU dataset in few-shot settings.

Dataset	Variation Type	Llama-2-7b-FLAN	Mistral-7B-FLAN
MMLU	Spelling Errors	$0.113_{\pm 0.116}$	$0.14_{\pm 0.143}$
	Prompt Templates	$0.229_{\pm 0.0.169}$	$0.668_{\pm 0.614}$
	Paraphrases	$0.163_{\pm 0.136}$	$0.187_{\pm 0.162}$
Alpaca	Spelling Errors	$0.317_{\pm 0.177}$	$0.284_{\pm 0.177}$
	Prompt Templates	$0.166_{\pm 0.129}$	$0.163_{\pm 0.145}$
	Paraphrases	$0.267_{\pm 0.192}$	$0.278_{\pm 0.2}$

Table 3: POSIX computed for Llama-2-7b-FLAN and Mistral-7B-FLAN (unlike the chat versions, these are only instruction-tuned on the FLAN dataset without any further RLHF).

templates. This might have given the FLAN-only models an edge over the chat versions.

5.3 Impact of Model Scale on Sensitivity

To study the effect of model scale on prompt sensitivity, we experiment with 1B and 7B variants of OLMo as well as 7B and 13B variants of Llama-2. Figure 3 and Figure 4 show the variations in the value of POSIX with model scale for prompt variants from *Mixture* of variation types, on both MMLU and Alpaca (Please refer to Appendix E for results on BBH). We observe that the 7B model is significantly more sensitive compared to the 1B model on the MMLU dataset; however, they are comparable in the case of the Alpaca dataset, in the case of OLMo. Similarly, even in the case of Llama-2, a 13B model is not guaranteed to always have lesser prompt sensitivity than a 7B model. This only re-emphasizes the fact that accuracy and sensitivity are two separate aspects — and higher accuracy does not necessarily imply better sensitivity and vice-versa. The POSIX values of OLMo-1B for all variation types are reported in Table 6 of Appendix C, and that of Llama-2-13B (base and chat) models are reported in Table 7 (for MMLU) and Table 8 (for Alpaca) of Appendix D.

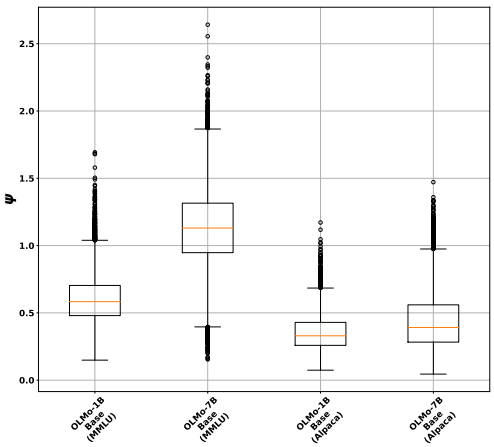


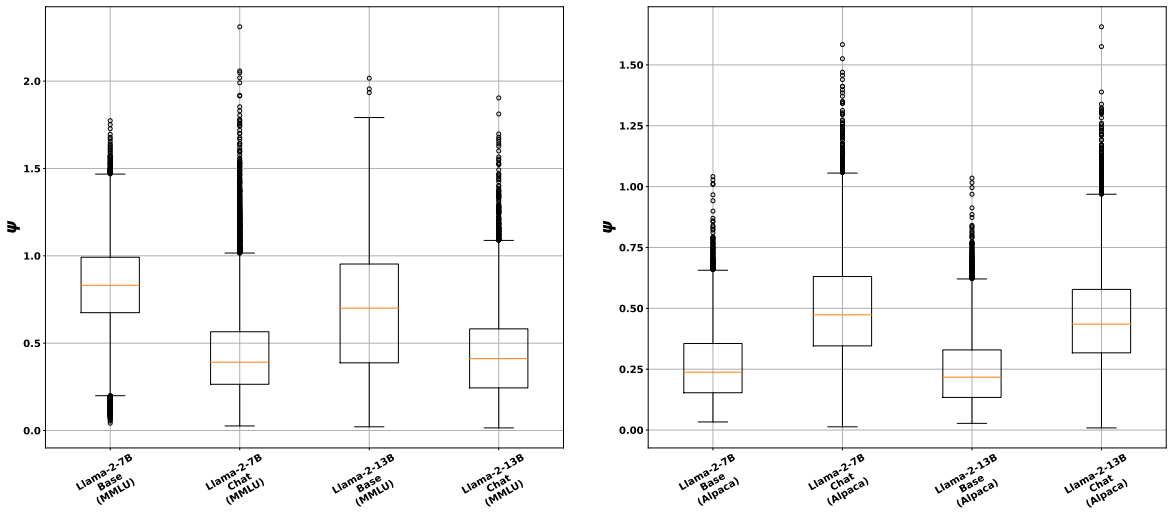
Figure 3: Box plots depicting distribution of $\psi_{\mathcal{M},\mathbf{x}}$ for two differently sized OLMo models (1B and 7B).

5.4 Few-shot Exemplars and Sensitivity

Table 2 consists of POSIX values computed for the few-shot setting in the case of the MMLU dataset. The key finding is that adding few-shot exemplars, even if it just a single example can significantly boost the robustness of LLMs towards variations in prompts. Although, adding even more few-shot examples might yield diminishing gains, i.e., when compared to the value that a single example adds, the additional value of a second or third few-shot exemplar is not that much — prompt sensitivity either remains about the same or slightly decreases.

5.5 Impact of Various Variation Categories

From Table 1, it can be observed that prompt template is the most sensitive variation type in the case of MCQs, and paraphrases are almost always the most sensitive variation type in the case of Alpaca (OLMo being the only exception). Moreover, the



(a) MMLU (MCQs)

(b) Alpaca (Open-ended generation)

Figure 4: Box plots depicting distribution of $\psi_{\mathcal{M},\mathbf{X}}$ for two differently sized Llama-2 models (7B and 13B).

sensitivity of the prompt template is more sensitive than any other perturbation in the case of Alpaca, thus making MCQ more sensitive overall (please refer to the *mixture* column in Table 1). Based on this prompt sensitivity analysis, we could offer some insights while performing prompt engineering as well — for MCQs, it is better to invest efforts in getting the proper prompt template while for open-ended questions, it is crucial to re-phrase the query properly.

6 Conclusions

We introduced POSIX - a novel prompt sensitivity index, as a reliable measure of sensitivity of LLMs towards intent-preserving variations in prompts such as spelling errors, prompt templates, and alterations in the wording. We presented thorough empirical analysis for the efficacy of POSIX in capturing prompt sensitivity and subsequently used it to measure and compare multiple open-source LLMs, revealing some interesting observations such as prompt template is the most sensitive variant type for MCQ tasks and paraphrasing is the most sensitive variant type for open-ended generation tasks, and also that parameter count or instruction tuning do not necessarily decrease prompt sensitivity of the models. These findings highlight the nuanced behaviour of LLMs towards prompt variations, underscoring the importance of considering prompt sensitivity index for their holistic evaluation.

Limitations

While POSIX has its own advantages, like the ability to work across different kinds of prompt variations and tasks, including open-ended generation with arbitrary response lengths, one of the main limitations of POSIX is its computational complexity. POSIX needs $\mathcal{O}(MN^2)$ log-likelihood comparisons if M is the total number of prompts in a dataset under consideration and N is the number of variations per prompt. Nevertheless, POSIX is very effective in incorporating various facets of prompt sensitivity.

Ethical Considerations

Since we use open-source large language models and open-source datasets like MMLU and Alpaca, our work encompasses all the corresponding considerations of those works. Although, our method would be expected to largely benefit the community by providing a reliable way to evaluate sensitivity of large language models towards variations in prompts. While attempting to paraphrase the prompts in MMLU using GPT-3.5-Turbo, quite a few prompts have been flagged as either violent or biased, etc. Most of them were from the *moral_scenarios* split of MMLU. We made sure to remove these from our analyses.

Acknowledgments

Tanmoy Chakraborty acknowledges the financial support of Adobe Faculty Award.

References

AI@Meta. 2024. Llama 3 model card.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *ArXiv preprint*, abs/2311.16867.

Simran Arora, Avanika Narayan, Mayee F. Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. 2022. Ask me anything: A simple strategy for prompting language models. *Preprint*, arXiv:2210.02441.

Greg Brooks, Tom Gorman, and Lesley Kendall. 1993. Spelling it out: the spelling abilities of 11- and 15-year-olds. Technical report, National Foundation for Educational Research (NFER), Slough.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric Xing, and Zhiting Hu. 2022. RLPrompt: Optimizing discrete text prompts with reinforcement learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3369–3391, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Avia Efrat, Or Honovich, and Omer Levy. 2023. LMentry: A language model benchmark of elementary language tasks. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10476–10501, Toronto, Canada. Association for Computational Linguistics.

Linton C Freeman. 1965. Elementary applied statistics: for students in behavioral science. *(No Title)*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. 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)*, pages 3816–3830, Online. Association for Computational Linguistics.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Olmo: Accelerating the science of language models. *Preprint*, arXiv:2402.00838.

Jiasheng Gu, Hongyu Zhao, Hanzi Xu, Liangyu Nie, Hongyuan Mei, and Wenpeng Yin. 2023. Robustness of learning from task instructions. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13935–13948, Toronto, Canada. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

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. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Alina Leidinger, Robert van Rooij, and Ekaterina Shutova. 2023. The language of prompting: What linguistic properties make a prompt successful? *Preprint*, arXiv:2311.01967.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby

Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2023. Holistic evaluation of language models. *Preprint*, arXiv:2211.09110.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 100–114, Dublin, Ireland and Online. Association for Computational Linguistics.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.

Sheng Lu, Hendrik Schuff, and Iryna Gurevych. 2023. How are prompts different in terms of sensitivity? *Preprint*, arXiv:2311.07230.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. 2024. State of what art? a call for multi-prompt llm evaluation. *Preprint*, arXiv:2401.00595.

Felipe Maia Polo, Ronald Xu, Lucas Weber, Mírian Silva, Onkar Bhardwaj, Leshem Choshen, Allysson Flavio Melo de Oliveira, Yuekai Sun, and Mikhail Yurochkin. 2024. Efficient multi-prompt evaluation of llms. *Preprint*, arXiv:2405.17202.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Laria Reynolds and Kyle McDonell. 2021. Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA. Association for Computing Machinery.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2023. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. *ArXiv preprint*, abs/2310.11324.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *ArXiv preprint*, abs/2206.04615.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2022. Selective annotation makes language models better few-shot learners. *Preprint*, arXiv:2209.01975.

Jiuding Sun, Chantal Shaib, and Byron C Wallace. 2024. Evaluating the zero-shot robustness of instruction-tuned language models. In *The Twelfth International Conference on Learning Representations*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *ArXiv preprint*, abs/2210.09261.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

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

Anton Voronov, Lena Wolf, and Max Ryabinin. 2024. Mind your format: Towards consistent evaluation of in-context learning improvements. *Preprint*, arXiv:2401.06766.

Lucas Weber, Elia Bruni, and Dieuwke Hupkes. 2023. The icl consistency test. *Preprint*, arXiv:2312.04945.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Qinyuan Ye, Maxamed Axmed, Reid Pryzant, and Fereshte Khani. 2024. Prompt engineering a prompt engineer. *Preprint*, arXiv:2311.05661.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *Preprint*, arXiv:2211.01910.

Kaijie Zhu, Jindong Wang, Jiaheng Zhou, Zichen Wang, Hao Chen, Yidong Wang, Linyi Yang, Wei Ye, Yue Zhang, Neil Zhenqiang Gong, and Xing Xie. 2023. Promptbench: Towards evaluating the robustness of large language models on adversarial prompts. *Preprint*, arXiv:2306.04528.

A Code and Data

Our code that we used for generating the variants of prompts and computing the sensitivity metric is open-sourced at https://github.com/kowndinya-renduchintala/POSIX. The code development utilized open-source tools, primarily relying on the HuggingFace library for inference, with PyTorch as the underlying framework. Both PyTorch and HuggingFace are licensed under permissive licenses, with PyTorch under the BSD license and HuggingFace under the Apache 2.0 license.

B Prompt Templates

The original template used for experiments on MMLU is:

`Q:{}\n(A){} (B){} (C){} (D){}\nA:`

The other 20 prompt templates used for the experiments with template variations on MMLU are listed in Table 4.

For experiments on the Alpaca dataset which contains open-ended questions, the default template used is: `Q:{}\nA:` . The other 20 prompt templates used for the experiments with template variations on Alpaca are listed in Table 5.

Seed	Prompt Template
0	`q:{}\n(A){} (B){} (C){} (D){}\na:`
1	`Q::{}\n(A){} (B){} (C){} (D){}\nA::`
2	`Q: {}\n(A){} (B){} (C){} (D){}\nA:`
3	`q:: {}\n(A){} (B){} (C){} (D){}\na::`
4	`Q::: {}\n(A){} (B){} (C){} (D){}\nA:::`
5	`Q: {} || (A){} (B){} (C){} (D){} || A:`
6	`q:::{}\n(A){} (B){} (C){} (D){}\na:::`
7	`Q: {}\n(A){} (B){} (C){} (D){}\nAnswer:`
8	`QUESTION:{}\n(A){} (B){} (C){} (D){}\nA:`
9	`Question:{}\n(A){} (B){} (C){} (D){}\nAnswer:`
10	`QUESTION:{}\n(A){} (B){} (C){} (D){}\nANSWER:`
11	`Question: {}\n(A){} (B){} (C){} (D){}\nAnswer:`
12	`Question::: {}\n(A){} (B){} (C){} (D){}\nAnswer:::`
13	`QUESTION: {}\n(A){} (B){} (C){} (D){}\nAnswer:`
14	`Question - {}\n(A){} (B){} (C){} (D){}\nAnswer -`
15	`question::{}\n(A){} (B){} (C){} (D){}\nanswer::`
16	`question: {}\n(A){} (B){} (C){} (D){}\nanswer:`
17	`Question: {} || (A){} (B){} (C){} (D){} || Answer:`
18	`QUESTION\t{}\n(A){} (B){} (C){} (D){}\nANSWER\t`
19	`Question: {} , (A){} (B){} (C){} (D){} , Answer:`

Table 4: Prompt Templates used for MMLU.

Also, for experiments on MMLU, we prepend the following instruction before the template:

`The following are multiple choice questions (with answers) about {subject}.\n\n`

The `{subject}` is filled with the corresponding topic name from MMLU.

Seed	Prompt Template
0	`q:{}\na:`
1	`Q::{}\nA::`
2	`Q: {}\nA:`
3	`q:: {}\na::`
4	`Q::: {}\nA:::`
5	`Q: {} || A:`
6	`q:::{}\na:::`
7	`Q: {}\nAnswer:`
8	`QUESTION:{}\nA:`
9	`Question:{}\nAnswer:`
10	`QUESTION:{}\nANSWER:`
11	`Question: {}\nAnswer:`
12	`Question::: {}\nAnswer:::`
13	`QUESTION: {}\nAnswer:`
14	`Question - {}\nAnswer -`
15	`question::{}\nanswer::`
16	`question: {}\nanswer:`
17	`Question: {} || Answer:`
18	`QUESTION\t{}\nANSWER\t`
19	`Question: {} , Answer:`

Table 5: Prompt Templates used for Alpaca.

C Sensitivity of OLMo-1B

Table 6 reports the POSIX values of OLMo-1B model for different variation types on MMLU and Alpaca.

Variation Type	MMLU-ZeroShot	Alpaca-ZeroShot
Spelling Errors	$0.089_{\pm 0.099}$	$0.257_{\pm 0.229}$
Prompt Templates	$0.896_{\pm 0.205}$	$0.355_{\pm 0.069}$
Paraphrases	$0.102_{\pm 0.082}$	$0.185_{\pm 0.128}$
Mixture	$0.602_{\pm 0.17}$	$0.358_{\pm 0.139}$

Table 6: Sensitivity of OLMo-1B.

D Sensitivity of Llama-2-13B

Table 7 reports the POSIX values of Llama-2-13B models (base and chat variants) for different variation types on MMLU and Table 8 reports it for Alpaca dataset.

Variation Type	Llama-2-13B	Llama-2-Chat-13B
Spelling Errors	$0.057_{\pm 0.073}$	$0.072_{\pm 0.086}$
Prompt Templates	$1.16_{\pm 0.518}$	$0.858_{\pm 0.324}$
Paraphrases	$0.093_{\pm 0.106}$	$0.095_{\pm 0.102}$
Mixture	$0.677_{\pm 0.349}$	$0.438_{\pm 0.248}$

Table 7: Sensitivity of Llama-2-13B (MMLU-ZeroShot)

E Sensitivity on the BBH dataset

Table 9 reports the POSIX values on the Big-Bench Hard (BBH) dataset. Additionally, Figure 5 depicts prompt sensitivity of various models on BBH dataset in the form of box plots and Figure 6 depicts

Variation Type	Llama-2-13B	Llama-2-Chat-13B
Spelling Errors	$0.141_{\pm 0.126}$	$0.222_{\pm 0.172}$
Prompt Templates	$0.177_{\pm 0.109}$	$0.134_{\pm 0.135}$
Paraphrases	$0.225_{\pm 0.173}$	$0.592_{\pm 0.3}$
Mixture	$0.249_{\pm 0.148}$	$0.461_{\pm 0.213}$

Table 8: Sensitivity of Llama-2-13B (Alpaca-ZeroShot)

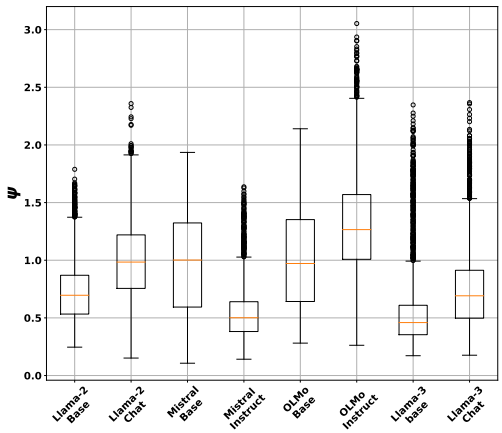


Figure 5: Box plots depicting distribution of $\psi_{\mathcal{M},\mathbf{x}}$ for various 7B models on Big Bench Hard (BBH) dataset

the effect of varying model scale on prompt sensitivity for BBH dataset. Please note that we only consider one kind of variation - prompt template - for this dataset. This is because in the case of BBH, many tasks like boolean_expressions, date understanding, geometric_shapes or dyck_languages, even minor spelling errors are not intent-preserving and paraphrasing would not be possible in many cases such as numerical expressions. For this dataset, we sample 2700 samples randomly from 23 tasks.

Model	BBH-ZeroShot
Llama-2-7b	$0.729_{\pm 0.295}$
Llama-2-7b-chat	$0.989_{\pm 0.351}$
Llama-3-8b	$0.58_{\pm 0.385}$
Llama-3-8b-chat	$0.745_{\pm 0.35}$
Mistral-7B	$0.966_{\pm 0.447}$
Mistral-7B-Instruct	$0.542_{\pm 0.239}$
OLMo-7B-Base	$1.029_{\pm 0.427}$
OLMo-7B-Instruct	$1.304_{\pm 0.47}$
OLMo-1B	$1.021_{\pm 0.292}$
Llama-2-13b	$0.851_{\pm 0.357}$
Llama-2-13b-chat	$0.972_{\pm 0.388}$

Table 9: Sensitivity of various models on the BBH dataset

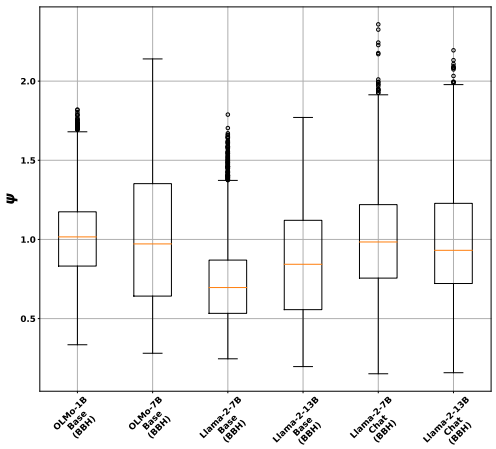


Figure 6: Box plots depicting distribution of $\psi_{\mathcal{M},\mathbf{x}}$ when varying model scale, for OLMo (1B and 7B) and Llama-2 (7B and 13B) – both base and chat variants.

F Examples of generated paraphrases

Please note that we open source all the data at https://github.com/kowndinya-renduchintala/POSIX. Although, following are some examples of generated paraphrases for few open-ended questions from alpaca data:

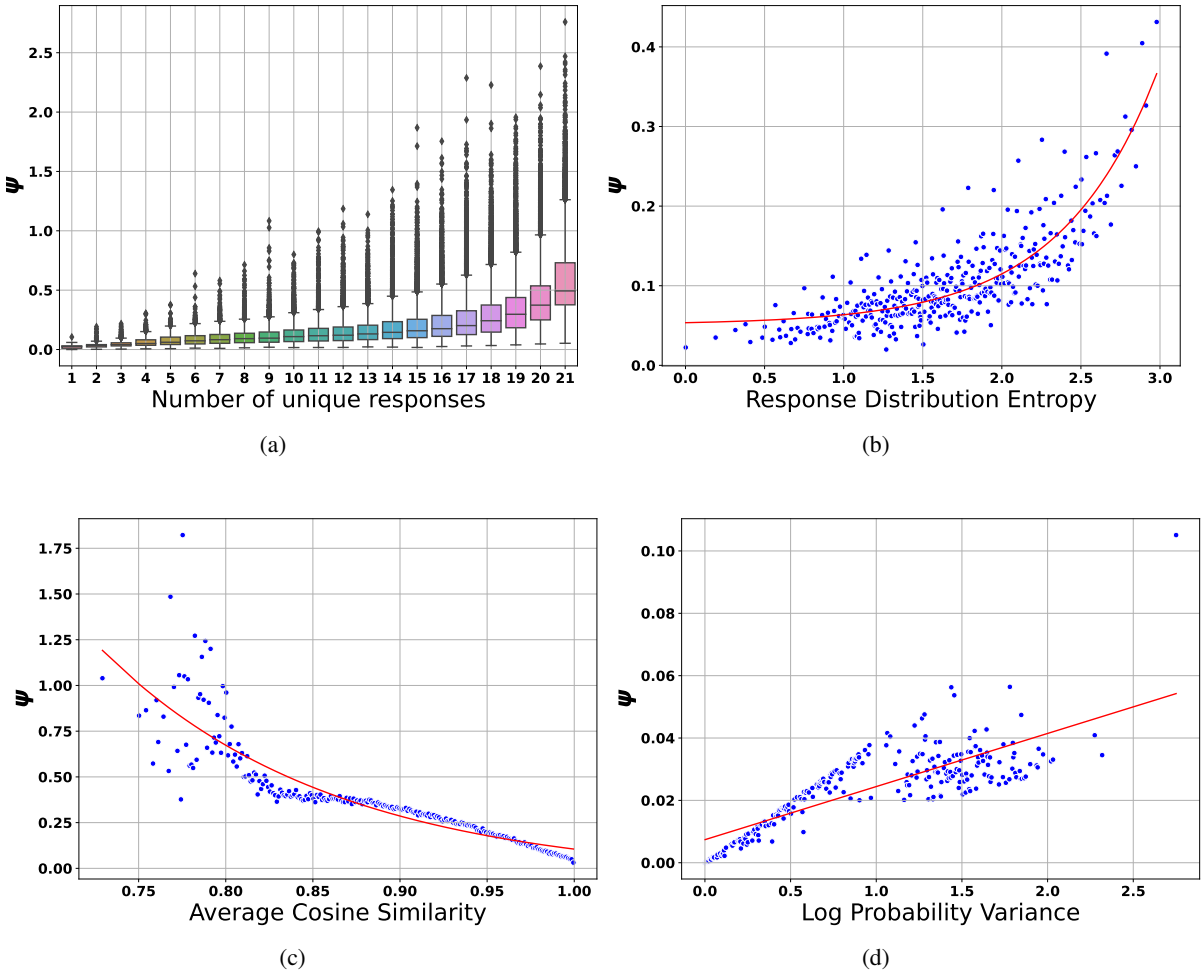
- **Original Question:** How much do you know about Buddhism?
  - What is your awareness of Buddhism?
  - What is your level of expertise on Buddhism?
  - Are you familiar with the principles of Buddhism?
  - What is your level of familiarity with Buddhism?
  - Can you share your knowledge of Buddhism with me?
- **Original Question:** Explain the concept of cognitive biases.
  - Interpret the idea of cognitive biases
  - Expound on the concept of cognitive biases
  - Elaborate on the concept of cognitive biases
  - Explicate the concept of cognitive biases
  - Spell out the notion of cognitive biases
- **Original Question:** Describe the best way to store fresh berries.
  - Provide instructions on how to store fresh berries for maximum freshness.
  - Offer advice on how to best store fresh berries.
  - Elaborate on the best way to store fresh berries to maintain their freshness.
  - Detail the optimal way to keep fresh berries fresh for longer.
  - Elaborate on the proper way to store a bunch of fresh berries.

G Efficacy of POSIX for Open-ended Generation

Figure 7 shows the correlation of POSIX with the four factors listed in Section 1 for a combination of all types of prompt variations in Alpaca, depicting the effectiveness of POSIX in successfully capturing the nuances of prompt sensitivity.

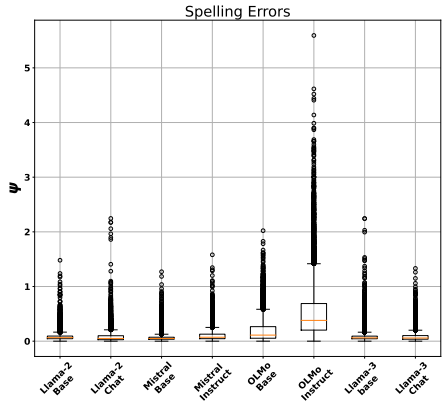
H Distribution of POSIX Values for All Variation Types

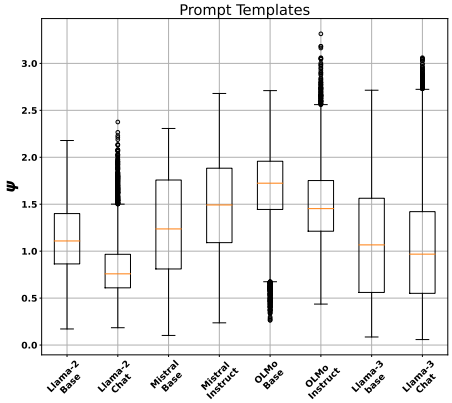
Figure 8 and Figure 9 depict the distribution of the values of POSIX for all models and variation types in MMLU and Alpaca, respectively.

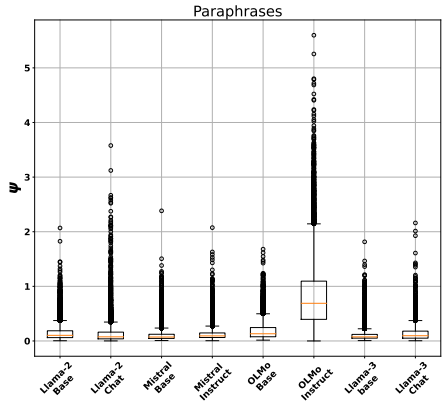


(a) (b) (c) (d)

Figure 7: Correlation plots of ψ with each of the four factors described in Section 3.3 in the case of Alpaca: (a) Response Diversity; (b) Response Distribution Entropy; (c) Semantic Coherence; (d) Variance in Confidence.







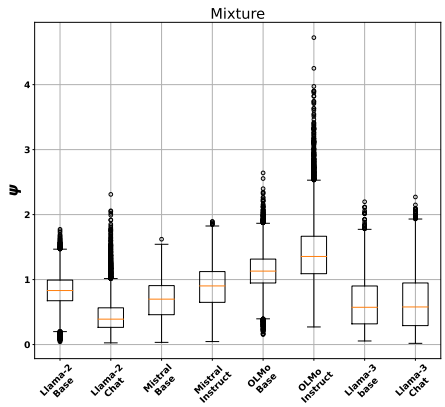
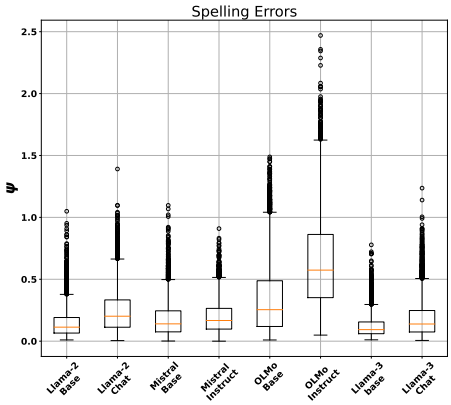
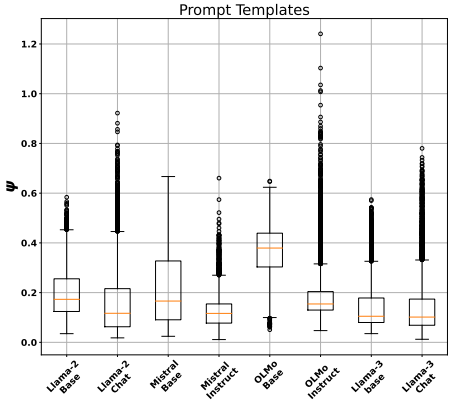
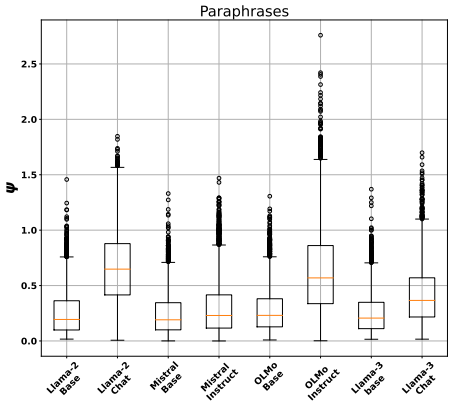


Figure 8: Box Plots depicting variation of ψ for different prompt variations in case of MMLU.







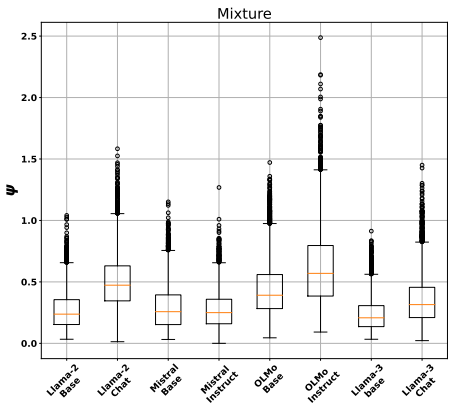


Figure 9: Box Plots depicting variation of ψ for different prompt variations in case of Alpaca.