Language Model Probabilities are *Not* Calibrated in Numeric Contexts

Charles J. Lovering[♡]
Michael Krumdick[♡] **Viet Dac Lai**[♠†] **Varshini Reddy**[♡]
Seth Ebner[♡] **Nilesh Kumar**[♣†] **Rik Koncel-Kedziorski**[◇†] **Chris Tanner**[♡]
[♡] Kensho Technologies [♠]Adobe [♣] RIT [◇]Apple
[†] Previously at Kensho Technologies
`charles.lovering@kensho.com`

Abstract

Some statements have one well-defined continuation (e.g., "the Eiffel Tower is in [*Paris*]"), whereas others have a natural distribution over multiple options (e.g., "the weighted coin flip was [*Heads/Tails*].") We argue that language model (LM) outputs should capture these natural distributions. Our work specifically tests whether LM output probabilities are *calibrated* to numeric information within their textual contexts. For example, if the context (the prompt) concerns two equally likely options (e.g., heads or tails for a fair coin), the LM output probabilities should also be equal. Likewise, in a context with nonuniformly likely events (e.g., rolling a pair with two dice) an LM should output proportionate probabilities. However, we find that even in simple settings, the best LMs (1) are poorly calibrated and (2) have systematic biases: artifacts like word identity, word order, and word frequency all impact calibration. For example, `gpt-4o-mini` often picks the first of two options presented in the prompt regardless of the options' implied likelihoods, whereas `Llama-3.1-8B` picks the second. Models do not allocate probability mass *among* valid options in a calibrated manner.

1 Introduction

We investigate if language model (LM) outputs are calibrated to the numeric content of their contexts. Consider the context below:

(1) From **98** *blue* marbles and **99** *red* marbles, Tommy reached blindly into a bag and grabbed a marble the color [*blue*/*red*]

This work tests the normative assumption that model probabilities for the next generated token should be calibrated to the relevant numeric content. For example, in Figure 1, the probability that an LM continues with one of the *bracketed tokens* should be proportional to the respective **bolded** values. If the context defines or implies a probability to hold within the context, then subsequent text generation should condition on that probability.

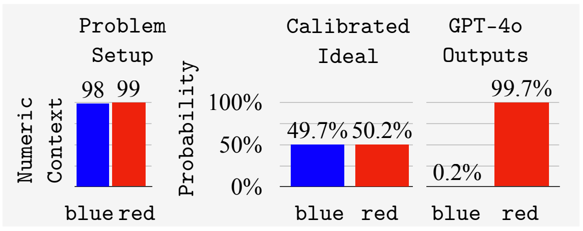

Figure 1: **Models are un-calibrated**. In this example, `gpt-4o` over-weights the option with a higher count of items beyond the calibrated probability, predicting *red* with 99.7% probability when 50.2% is appropriate. We find consistent patterns of uncalibrated behavior.

What Do We Mean By Calibration?

LMs' output probabilities are *calibrated* to their contexts if the probabilities of relevant tokens correspond to the numeric content implicitly or explicitly defined in those contexts. If tokens $t_1, t_2, \ldots, t_n$ are indicated by context C to have probabilities $P = p_1, p_2, \ldots, p_n$, a calibrated LM m outputs corresponding probabilities $\pi_i = m(t_i|C)$:

$$p_i \propto \pi_i, \quad i \in 1, 2, \ldots, n.$$

We set calibration as the distance between P and $\Pi = \pi_1, \pi_2, \ldots, \pi_n$ (Section 3.2).

Calibration within numeric contexts is a simple concept, but there is good reason to believe that models may struggle with it. Numbers within large pre-training datasets do not all occur with identical frequencies (e.g., numbers ending in 5 appear more than numbers ending in 7), as shown by Zhou et al. (2023) for the Pile dataset and our findings with Dolmo (Soldaini et al., 2024) (Section 6). These differences may have downstream effects on LMs' biases across different numbers, making it difficult for language models to exhibit calibrated outputs.

Model calibration is related to models' mathematical capabilities. If a model struggles to answer basic math questions, it may be difficult for the model to be properly calibrated. Although models are quickly improving at mathematical reasoning (Lin et al., 2024; Shao et al., 2024; MistralAI, 2025; Chervonyi et al., 2025), our work demonstrates that models are not well calibrated even in simple scenarios that require little reasoning. This may be because the difficulty lies not in the underlying mathematical ratios but in connecting these implicit probabilities with their explicit probabilities through a softmax. Meister et al. (2024) shows that models struggle to directly model distributions.

Because LMs are increasingly used, calibrating LM outputs is increasingly important. Notably, sampling is still often used to increase the diversity of model outputs (Chen et al., 2021); if models were only used greedily studying model logits would no longer be of direct interest. Not producing a calibred result may have no impact over a single interaction because any given outcome is compatible with a calibrated distribution (assuming some probability mass on all options).

However, failed calibration may have an impact across multiple users or repeat interactions. For example, an uncalibrated model may always recommend visiting a particular restaurant despite the city having hundreds of equally good options. This model's calibration failure could result from something as idiosyncratic as the restaurant's name. In this scenario, the uncalibrated model negatively impacts many parties (e.g., the other restaurants, customers, etc). This restaurant example is simply one of the many scenarios where users rely on accurate and helpful recommendations (e.g., for movies, job candidates, vacations, colleges, etc). Having correctly calibrated models is critical for situations involving medical or health services, such as making radiology predictions (Shreekumar, 2025).

We find that many LMs are poorly calibrated in numeric contexts, failing in systematic ways. Like in Example 1, some models pick options based on position (e.g., always the first option). Other models overly pick the option with the higher number. Word identity of the options and the vocabulary frequency of the numeric information appear to interact with these biases and, thus, the calibration. Our work demonstrates that LMs are consistently biased suggesting that there is risk to using LMs for probabilistic outcomes.

Contributions

1. Instruction-tuning helps by increasing the model's cumulative probability for all relevant options. Yet, models generally remain poorly calibrated – sometimes worse than simple baselines.

2. Models exhibit mode collapse: instruction-tuning overly reduces the entropy of the output distribution.

3. Models exhibit systematic biases over word identity and order, often overriding the numeric information in the context.

4. Frequency effects of the number tokens in the training data impact calibration.

2 Related Work

Models struggle to calibrate internal uncertainty with textual outputs. A related line of work studies how well models calibrate prediction confidence with prediction accuracy: a calibrated model with 0.80 confidence on each prediction should classify 80% of examples correctly (Guo et al., 2017; Minderer et al., 2021). Under this definition, Wei et al. (2024) reports GPT models have good calibration, whereas, Phan et al. (2025) reports the opposite. Another line of work investigates how well model outputs align with uncertainty (Yona et al., 2024; Kumar et al., 2024; Zhou et al., 2023; Lin et al., 2022), termed linguistic calibration (Mielke et al., 2022). Yona et al. (2024) reports that models struggle to exhibit internal uncertainty in text. Kumar et al. (2024) measures how well confidence is aligned internally (model probabilities) and externally (via eliciting a model to choose among a Likert scale). In our work, we look at the agreement between the probability of predicted outputs and the contextually defined likelihood of outputs in the text. We find that models are often internally *certain* when they should be *uncertain*; this mode collapse arises both via supervised fine-tuning and reward-based fine-tuning (O'Mahony et al., 2024; Janus, 2022). Zhou et al. (2024) also finds that reward-based fine-tuning biases models towards certainty, generating strengtheners (e.g., "I am certain…") over weakeners (e.g., "Maybe…"). Notably, faithful calibration is not always desirable; to simplify our experiments, we set aside phenomena like hyperbole and pragmatic effects (Tsvilodub et al., 2025).

Recent work finds that language models struggle to simulate randomness: coin flip predictions are biased in favor of heads and the first outcome mentioned in context (Van Koevering and Kleinberg, 2024). Hopkins et al. (2023), similar to our work, shows that LMs struggle to induce uniform distributions in the 0 to 1 interval. Recentlt, Meister et al. (2024) shows that models struggle to directly model distributions and Gupta et al. (2025) again showed that models have strong priors in probabilistic settings (coin flips).

Biases in language models. There are large-scale efforts to remove biases from models (Rudinger et al., 2018; Bai et al., 2022). Pezeshkpour and Hruschka (2024) shows that LMs prefer multiple-choice options based on their position. Using methods similar to our's, Kusner et al. (2017) studies probability differences in the setting of counterfactual fairness. We find evidence of similar biases impacting calibration.

3 Experimental Design

3.1 Problem Setup

We introduce three templated datasets. Each instance is a context C to be continued by a relevant token among $t_1, t_2, \ldots, t_n = T \subset V$, where V is the full vocabulary. Each token t_i is associated with an implied probability p_i, forming a distribution over T, $P = p_1, p_2, \ldots, p_n$. In Example 1, T is $\{\textit{red}, \textit{blue}\}$ with $P = \{\textit{red} : {}^{98}/_{197}, \textit{blue} : {}^{99}/_{197}\}$. For a model m, we define $\Pi = \pi_1, \pi_2, \ldots, \pi_n$ where $\pi_i \approx m(t_i|C)$, with one modification. We sum the probabilities of common tokenizations for a given word, namely, capitalization and spaces: $\pi_i = \sum_{s \in \text{Tokenizations}(t_i)} m(s|C)$. For example, for the option *red*, we sum probabilities for "red", "Red", "_red", and "_Red." This approach does not take into account there are many grammatically ways of continuing the sentence. However, empirically, instruction-tuned models put almost all probability mass on T. This imperfect approach still allows us to meaningfully study the problem.[1]

3.2 Metrics and Evaluation

We use three metrics to test if models calibrate to numeric contexts and where they go wrong.

Probability Mass (PM): A model's calibration, the relative probability mass across tokens in *T*, is well-founded only if there is sufficient probability mass upon the tokens in T. We measure this directly with PM, PM$(T) \doteq \sum_{t \in T} \pi_t$. If PM$(T)$ for a model is low (say, 0.30), then it is unclear if the model is capturing the intended relationship between the context and T. While meaningful distribution across T could still exist if these values were normalized, we set this case aside. When PM is high (say, > 0.75), we can start to meaningfully ask questions about how probability mass is allocated among, say, *red* and *blue*.

Wasserstein Distance (WD): To measure calibration, we use WD (Kantorovich, 1939), which captures the movement between one distribution (or set of values) and another.[2] We use SciPy's `wasserstein_distance_nd` implementation.[3]

Relative Entropy (RE): To help understand a model's behavior, we propose RE $= \mathrm{H}(\Pi) - \mathrm{H}(P)$, the difference in entropy H between model and ideal calibrated probabilities. RE < 0 means the model's probabilities are overly concentrated and RE > 0 means they are overly diffuse.

3.3 Models

Our experiments assume access to model logits. We test range of open-source models with both base and instruction-tuned versions available: `Mistral-7B-v0.1`, `Mistral-7B-v0.3`, Mixtral-8x7B-v0.1, `Yi-1.5-9B`, `Yi-1.5-34B`, `Llama-3.1-8B`, `gemma-2-9b`, `gemma-2-27b` (Jiang et al., 2023, 2024; Young et al., 2024; Dubey et al., 2024; Team et al., 2024); and four proprietary models, `gpt-3.5`, `gpt-4-turbo`, `gpt-4o-mini`, and `gpt-4o` (Brown et al., 2020; Achiam et al., 2023; OpenAI, 2024). We prompt instruction-tuned models to calibrate outputs to the numbers in context. We refer to the base version of the models as "Base" and the instruction-tuned, prompted models as "Chat."

3.4 Datasets

We introduce three datasets *colors*, *wordproblems*, and *distributions*. The following are representative examples, respectively: [4]

[1]See Appendix A for details on reproducibility and `https://drive.google.com/drive/folders/11mY5vbNanqMA2wwci84I5cipwh7FPPPB` for code/results.

[2]Kullback–Leibler (KL) divergence is also a natural choice. However, some problems in our datasets include options with the value zero where KL would be undefined. We also compute Mean-Squared Error (MSE) but don't find a material difference in the findings so we only report the WD.

[3]`https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wasserstein_distance_nd.html`

[4]Our datasets test basic ratios and probability. As models improve, it would be exciting if datasets targeted concepts like

(2) From **17** *red* marbles and **99** *blue* marbles, Billy reached blindly into the bag and grabbed a marble with the color [*red*/*blue*]

(3) There was a grove with **17** *spruce* and **99** *cedar*. During a thunderstorm lightning struck a tree. All the trees were about the same height and elevation. Turned out, the species of the tree was [*spruce/cedar*]

(4) Sampling from a uniform distribution with support from **2** inclusive to **5** exclusive, I sampled the number [*0*/*1*/***2***/***3*** . . . *9*]

Table 1 reports the number of problems per dataset. For **(a)** *colors*, we use 5 templates, 3 numeric scales with 100 configurations each, and 110 permutations of color options (e.g., red/blue and blue/red).[5] The numeric scales are (1) numbers 1 to 10, (2) ten numbers sampled from 10 to 100, and (3) ten numbers sampled from 100 to 999. Pilot experiments with larger numbers (> 1000) did not see marked differences in calibration. Still, future work should study a wider range of numbers. For **(b)** *wordproblems*, there are 10 templates, the same 3 numeric scales with 100 configurations each, and 4-10 pairs of options (e.g., *spruce/cedar*, or *nurse/doctor*) depending on the template. For every pair of options in *colors* and *wordproblems*, we include all combinations of numbers creating a balanced set of examples: the first number is smaller in half of the examples and larger in the rest. For **(c)** *distributions*, the configuration is different. The task is to pick a number uniformly from an interval. The valid options in T are all integers in the interval. There are 5 templates, different range inclusivities, and 320 pairs of numbers that define the intervals ranging from 10 to 1000. Each problem is set to occur within an interval length of less than 10 and requires only a single token prediction.

When we compute standard errors we cluster option pairs together (Miller, 2024). All our results show the 95% confidence interval (not the standard error) either explicitly or using $\pm$ notation.

3.5 Reference Behaviors and Baselines

Reference Behaviors. Understanding a model's behavior across many instances can be difficult, but we observe that models often produce one of six different patterns of behavior, and as such, we categorize model behavior into these as references.

Summaries are limited: `gpt-4o-mini`, when the color *white* is listed second in the prompt, tends to pick the color listed first. This is classified as Pick First but could be "pick the option not starting with *w*." These behaviors are compatible with model outputs over many examples and settings, but are descriptive, not causal. Also note that over one example there are multiple valid explanations (e.g. in Example 1 could be Pick Higher or Pick Lower). Thus, we classify behavior over sets of examples balanced over the numeric values.

0. `Null`: PM (see below) is closer to 0 than other behaviors.
1. `Calibrated`: Sets $\Pi = P$, the best case.
2. `Pick Higher`: Sets all probability mass on $\arg\max P$.
3. `Pick lower`: Sets all probability mass on $\arg\min P$.
4. `Pick First`: Sets all probability mass on the first option, ignoring numeric information.
5. `Pick Second`: analogously sets all probability mass on the second option.

Baselines. We introduce two more baselines besides using the reference behaviors above as baselines. First, `Pick Higher`$_{p=q}$ places q (say, 0.7) on $\arg\max P$ and the remainder of the probability mass evenly across other tokens in T. Second, `Random` baselines to contextualize our results. For every problem in the dataset, the random baseline Random$_\tau$ produces a Π by uniformly sampling two random numbers from 0 to 1, which are then transformed into a distribution parameterized by a temperature τ using the softmax function.

Dataset	# Problems	# Option Sets	Entropy (bits)
colors	165.0K	55 (110)	$0.81_{0.06}$
wordproblems	33.6K	55 (110)	$0.81_{0.06}$
distributions	4.5K	320	$2.27_{0.17}$

Table 1: **Datasets.** The first two datasets have 55 pairs of options; 110 when order is considered. For the the third, there are 320 pairs of numbers that define the distribution intervals. The first two datasets use the same underlying numbers, sharing the same average entropy per problem instance. The entropy is shown as an average and sub-scripted by the variance across problem instances. *Distributions* problems are across more options (2+) and expect a uniform distribution.

Bayes' theorem, De Morgan's laws, event independence, etc.

[5]Calibration could be well-studied by sampling non-exhaustively across these permutations but doing so lets us highlight biases across different options.

Example with Baseline Scores. Table 2 provides baseline scores for a single example (numbers 51 and 98) to provide reference metric scores. Beyond showing how WD scores range, this example again highlights how different baselines (say, Pick First and Pick Higher) will have the same behavior for some instances; references are only delineated over a pool of examples were the behaviors are mutually exclusive. See Appendix D to see how different models behave on this example.

	$P(t_1)$	$P(t_2)$	PM	RE	WD
Ideal	0.34	0.66	1.00	1.00	0.00
Null	0.00	0.00	0.00	*n/a*	0.74
Pick First	1.00	0.00	1.00	-0.93	0.93
Pick Lower	1.00	0.00	1.00	-0.93	0.93
Pick Second	0.00	1.00	1.00	-0.93	0.48
Pick Higher	0.00	1.00	1.00	-0.93	0.48
Pick Higher$_{p=0.7}$	0.30	0.70	1.00	-0.05	0.06
Pick Higher$_{p=0.6}$	0.40	0.60	1.00	0.04	0.08
Pick Higher$_{p=0.8}$	0.20	0.80	1.00	-0.21	0.20
Pick Higher$_{p=0.9}$	0.10	0.90	1.00	-0.46	0.34

Table 2: Each row shows the performance of the given baseline an example with numbers 51 and 98. This table is meant to be a helpful reference to build an intuition about performance score magnitudes.

4 Results

4.1 Probability Mass

First, we closely examine the probability mass (PM) on the relevant tokens. Consider Example 1 where `gpt-4o` is to pick either *Red* or *Blue*. There, a high PM means that the model successfully allocated probability mass over these tokens. Success depends on the design of our problem scenarios and model capability. The high scores suggest that the models correctly interpret the basic task.

Table 3 reports PM for the base and chat versions of all tested models. **Instruction-tuned models have statistically significantly higher PMs over their base versions.** This can be seen by the strictly positive 95% confidence intervals of the difference between the chat and base scores in the rightmost column. Except for `Llama-3.1-8B` and `Yi-1.5-34B` on *distributions*, this holds for all models. The instruction-tuned PM scores are all reasonably high, suggesting that the datasets' sentence templates are well-formed. The primary exception is `Mistral-7B-v0.1`. Notably, `gpt-*` models had lower PM on *wordproblems*.

	Base PM	Chat PM	Chat - Base Δ (95% CI)
colors			
Llama-3.1-8B	$0.38_{\pm 0.01}$	$0.80_{\pm 0.00}$	(+0.41, +0.43)
Mistral-7B-v0.1	$0.30_{\pm 0.01}$	$0.54_{\pm 0.01}$	(+0.22, +0.25)
Mistral-7B-v0.3	$0.33_{\pm 0.01}$	$0.76_{\pm 0.02}$	(+0.42, +0.44)
Mixtral-8x7B	$0.36_{\pm 0.01}$	$0.99_{\pm 0.00}$	(+0.62, +0.63)
Yi-1.5-9B	$0.31_{\pm 0.01}$	$0.99_{\pm 0.00}$	(+0.67, +0.70)
Yi-1.5-34B	$0.42_{\pm 0.01}$	$0.62_{\pm 0.02}$	(+0.17, +0.22)
gemma-2-9b	$0.39_{\pm 0.01}$	$0.99_{\pm 0.00}$	(+0.59, +0.61)
gemma-2-27b	$0.54_{\pm 0.01}$	$1.00_{\pm 0.00}$	(+0.45, +0.46)
gpt-3.5	-	$1.00_{\pm 0.00}$	-
gpt-4-turbo	-	$1.00_{\pm 0.00}$	-
gpt-4o-mini	-	$1.00_{\pm 0.00}$	-
gpt-4o	-	$1.00_{\pm 0.00}$	-
wordproblems			
Llama-3.1-8B	$0.54_{\pm 0.05}$	$0.86_{\pm 0.03}$	(+0.25, +0.38)
Mistral-7B-v0.1	$0.44_{\pm 0.04}$	$0.69_{\pm 0.03}$	(+0.22, +0.28)
Mistral-7B-v0.3	$0.42_{\pm 0.04}$	$0.84_{\pm 0.06}$	(+0.33, +0.51)
Mixtral-8x7B	$0.57_{\pm 0.03}$	$0.97_{\pm 0.02}$	(+0.37, +0.44)
Yi-1.5-9B	$0.58_{\pm 0.03}$	$0.88_{\pm 0.03}$	(+0.25, +0.36)
Yi-1.5-34B	$0.53_{\pm 0.05}$	$0.72_{\pm 0.06}$	(+0.14, +0.25)
gemma-2-9b	$0.63_{\pm 0.04}$	$0.98_{\pm 0.01}$	(+0.31, +0.40)
gemma-2-27b	$0.59_{\pm 0.03}$	$1.00_{\pm 0.00}$	(+0.38, +0.45)
gpt-3.5	-	$0.60_{\pm 0.05}$	-
gpt-4-turbo	-	$0.46_{\pm 0.07}$	-
gpt-4o-mini	-	$0.55_{\pm 0.05}$	-
gpt-4o	-	$0.60_{\pm 0.05}$	-
distributions			
Llama-3.1-8B	$0.82_{\pm 0.00}$	$0.78_{\pm 0.01}$	(-0.06, -0.04)
Mistral-7B-v0.1	$0.92_{\pm 0.00}$	$0.95_{\pm 0.00}$	(+0.03, +0.03)
Mistral-7B-v0.3	$0.91_{\pm 0.00}$	$1.00_{\pm 0.00}$	(+0.09, +0.09)
Mixtral-8x7B	$0.96_{\pm 0.00}$	$1.00_{\pm 0.00}$	(+0.03, +0.04)
Yi-1.5-9B	$0.90_{\pm 0.00}$	$0.96_{\pm 0.01}$	(+0.06, +0.07)
Yi-1.5-34B	$0.95_{\pm 0.00}$	$0.91_{\pm 0.02}$	(-0.06, -0.02)
gemma-2-9b	$0.84_{\pm 0.01}$	$0.97_{\pm 0.01}$	(+0.13, +0.15)
gemma-2-27b	$0.96_{\pm 0.00}$	$1.00_{\pm 0.00}$	(+0.03, +0.04)
gpt-3.5	-	$0.99_{\pm 0.00}$	-
gpt-4-turbo	-	$1.00_{\pm 0.00}$	-
gpt-4o-mini	-	$0.68_{\pm 0.01}$	-
gpt-4o	-	$0.95_{\pm 0.01}$	-

Table 3: **Probability Mass Results** for the base and chat versions of models and the 95% confidence intervals of their paired difference. Statistically significant increases in PM are highlighted blue (better performance); decreases are highlighted orange (worse performance).

4.2 Calibration

Next, we examine the calibration scores of instruction-tuned models in Table 4. Given the low PM scores of the base models, we leave those results to Appendix Table 11. Our high-level finding is simple: all tested models are poorly calibrated, and their performance is typically worse than simple baselines. Some models are mildly better than a baseline that sets 100% probability to the option associated with the higher number, but most are not. Another baseline, Pick Higher$_{p=0.7}$, which allocates 70% probability to the option as-

	colors	*wordp.*	*distr.*
Pick Higher	$0.47_{\pm0.01}$	$0.44_{\pm0.00}$	-
Pick Higher$_{p=0.7}$	$0.15_{\pm0.01}$	$0.17_{\pm0.00}$	-
Pick Lower	$0.95_{\pm0.01}$	$0.98_{\pm0.00}$	-
Pick First/Second	$0.71_{\pm0.02}$	$0.71_{\pm0.00}$	$0.38_{\pm0.00}$
Random$_{\tau=0.01}$	$0.71_{\pm0.02}$	$0.69_{\pm0.00}$	$0.86_{\pm0.00}$
Random$_{\tau=1.00}$	$0.27_{\pm0.01}$	$0.29_{\pm0.00}$	$0.38_{\pm0.00}$
Llama-3.1-8B	$0.40^{♠}_{\pm0.01}$	$0.48_{\pm0.02}$	$0.43_{\pm0.01}$
Mistral-7B-v0.1	$0.50_{\pm0.01}$	$0.49_{\pm0.01}$	$0.22^{♣}_{\pm0.01}$
Mistral-7B-v0.3	$0.48_{\pm0.02}$	$0.55_{\pm0.01}$	$0.56_{\pm0.01}$
Mixtral-8x7B	$0.51_{\pm0.01}$	$0.60_{\pm0.03}$	$0.71_{\pm0.01}$
Yi-1.5-9B	$0.49_{\pm0.01}$	$0.56_{\pm0.01}$	$0.61_{\pm0.01}$
Yi-1.5-34B	$0.55_{\pm0.01}$	$0.57_{\pm0.01}$	$0.62_{\pm0.01}$
gemma-2-9b	$0.50_{\pm0.01}$	$0.52_{\pm0.02}$	$0.59_{\pm0.01}$
gemma-2-27b	$0.40^{♠}_{\pm0.00}$	$0.48_{\pm0.01}$	$0.59_{\pm0.01}$
gpt-3.5	$0.30^{♠}_{\pm0.01}$	$0.57_{\pm0.02}$	$0.42_{\pm0.01}$
gpt-4-turbo	$0.42^{♠}_{\pm0.01}$	$0.62_{\pm0.02}$	$0.69_{\pm0.01}$
gpt-4o-mini	$0.40^{♠}_{\pm0.01}$	$0.57_{\pm0.02}$	$0.57_{\pm0.01}$
gpt-4o	$0.40^{♠}_{\pm0.00}$	$0.57_{\pm0.02}$	$0.49_{\pm0.01}$

Table 4: **Calibration Results.** This table shows calibration scores (WD) for Chat models along with 95% confidence intervals (±). For *colors*, six models have a statistically significantly better calibration than Pick Higher, denoted ♠. For *wordproblems*, all models underperform the baselines. For *distributions*, only `Mistral-7B-v0.1` outperforms the Pick First baseline, ♣.

	colors	*wordp.*	*distr.*
Pick Higher$_{p=0.7}$	$0.02_{\pm0.01}$	$0.07_{\pm0.00}$	-
Pick Higher / Lower	$-0.86_{\pm0.01}$	$-0.81_{\pm0.00}$	-
Pick First / Second	$-0.86_{\pm0.01}$	$-0.81_{\pm0.00}$	$-2.26_{\pm0.01}$
Random$_{\tau=0.01}$	$-0.80_{\pm0.01}$	$-0.76_{\pm0.00}$	$-2.07_{\pm0.01}$
Random$_{\tau=1.0}$	$0.11_{\pm0.01}$	$0.16_{\pm0.00}$	$1.35_{\pm0.01}$
Llama-3.1-8B	$-0.11_{\pm0.02}$	$-0.33_{\pm0.06}$	$-0.70_{\pm0.03}$
Mistral-7B-v0.1	$-0.21_{\pm0.02}$	$-0.24_{\pm0.04}$	$0.05_{\pm0.02}$
Mistral-7B-v0.3	$-0.36_{\pm0.03}$	$-0.47_{\pm0.04}$	$-1.07_{\pm0.04}$
Mixtral-8x7B-v0.1	$-0.67_{\pm0.01}$	$-0.65_{\pm0.05}$	$-1.59_{\pm0.04}$
Yi-1.5-9B	$-0.55_{\pm0.01}$	$-0.55_{\pm0.03}$	$-1.13_{\pm0.05}$
Yi-1.5-34B	$-0.45_{\pm0.02}$	$-0.53_{\pm0.04}$	$-1.32_{\pm0.04}$
gemma-2-9b	$-0.44_{\pm0.01}$	$-0.45_{\pm0.03}$	$-1.11_{\pm0.04}$
gemma-2-27b	$-0.63_{\pm0.01}$	$-0.57_{\pm0.02}$	$-1.14_{\pm0.05}$
gpt-3.5	$-0.35_{\pm0.03}$	$-0.68_{\pm0.04}$	$-0.69_{\pm0.02}$
gpt-4-turbo	$-0.65_{\pm0.01}$	$-0.69_{\pm0.04}$	$-1.54_{\pm0.03}$
gpt-4o-mini	$-0.53_{\pm0.01}$	$-0.58_{\pm0.05}$	$-1.30_{\pm0.02}$
gpt-4o	$-0.61_{\pm0.00}$	$-0.63_{\pm0.04}$	$-0.92_{\pm0.02}$

Table 5: **Relative Entropy (RE) Results.** With the exception of `Mistral-7B-v0.1`, RE is below a calibrated level across all datasets.

sociated with the higher number, outperforms all models. This result, more than anything, describes the best average output probabilities for *colors* and *wordproblems*. This all goes to say: **Although the models can allocate probability mass on valid options, they cannot properly allocate probability mass *among* these options in a calibrated manner.** In Appendix B, we find evidence that our calibration results are stable across a broader range of templates and template variations.

4.3 Relative Entropy

The section above demonstrated that the models cannot allocate the probability mass *among* the valid options in a calibrated manner. Here, as shown in Table 5, we find that these models fail primarily by over-allocating probability mass on one of the two options: mode collapse. Every model's RE was statistically significantly less than the calibrated result with large effects on all datasets. Models with the best calibration scores (gpt-* models) also have the lowest relative entropy, suggesting that no model approaches well-calibrated behavior.

Instruction-tuning appears to cause this mode collapse. There is a reduction in entropy between the base and chat model versions, on average, 0.50/0.36/1.19 bits on the three datasets, respectively, corresponding to a 47/42/55% percent of the entropy of the ideal calibrated result. (See Appendix G for more details.)

5 Analysis: Option Identity and Order

In this section, we study the particular behaviors that models exhibit on the *colors* dataset, finding that **the models exhibit systematic biases and preferences based upon word identity and word order.** Understanding that (1) the models are poorly calibrated and (2) exhibit low entropy falls directly out of our proposed metrics. However, understanding how the models behave is more complicated. To start, we characterize the behavior of `gpt-4o-mini` in Figure 2 (more in Appendix H). Figure 3 shows model behavior across all settings. We define these behaviors in Section 3.5.

Beyond the specific systematic patterns we uncover, we want to stress that the existence of these patterns is important. Every use case of LMs that requires such choices may have similar patterns. **That these systematic errors arise even with innocuous features like color should give us pause about using models to make probabilistic choices in more complicated scenarios.**

Figure 2 captures model behavior across all tested pairs of color options. The most apparent note to make is the diagonal asymmetry. **This asymmetry shows a hierarchical, systematic bias across word identity (specific colors) and word order.** Each cell within the heatmap reports the rate at which the model's output probabilities are

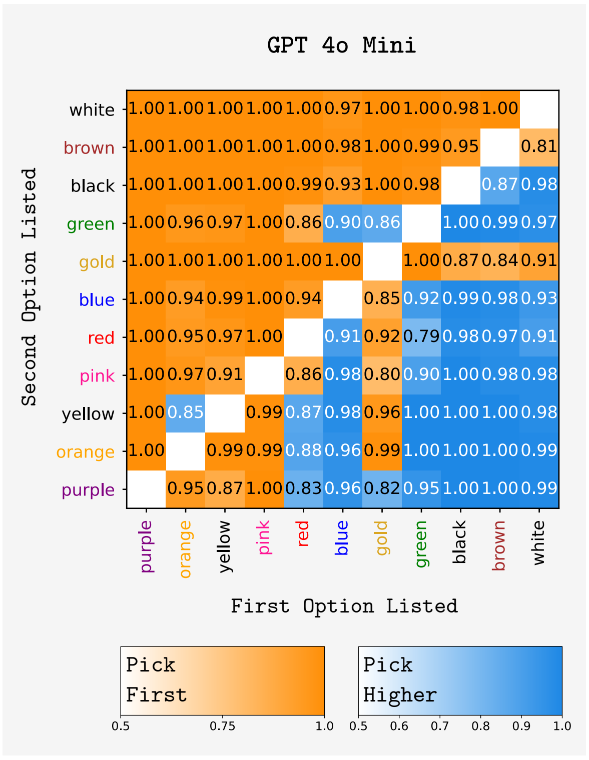

Figure 2: **Systematic Patterns in Model Behavior.** Each cell corresponds to model behavior across 100 examples. The number is the rate the outputs are compatible with the given behavior. For example, 1.00 in the first top-left cell means that for 100/100 instances a majority of the probability mass is on the first option. The top-left cell corresponds to instances where purple is first in the prompt and white is second. High rates across multiple behaviors are impossible; they are mutually exclusive across the 100 instances. See Figure 4 for a representative instance drawn from the top-left cell.

compatible with the given behavior across a full grid (set of combinations) of number pairs. For every number combination we include both orderings meaning that there are equal number of cases where the first number is higher as there where the second number is higher. The behavior with the highest compatibility rate is displayed and indicated by the color. For example, if the model had rates 100 Pick Higher, 50 Pick First, and 50 Pick Second, the plot would show the rate of 1.00 (100/100) for Pick Higher. A high rate of two behaviors cannot happen because the 100 examples are balanced over numeric values.

The behaviors exhibited across the diagonal are often different. This means that option order, listing a color first or second, greatly affects calibration. For example, in Figure 2, we can see that when "white" is mentioned first, the model tends to Pick Higher and when it is mentioned second, the model tends to Pick First. Because Pick Higher is a more calibrated behavior than Pick First, mentioning "white" first leads to lower calibration scores.

We measure these differences in calibration across options and models, more in Appendix H. Color orderings are correlated among similar models (`gpt-4o`/`gpt-4o-mini`), suggesting that training data help form these patterns of behavior.

Figure 3, reports the overall behavior rates for each model. Appendix H details how we compute the proportions of different behaviors across the dataset. `gpt-*` models (and `gemma-2-27b`) tend to `Pick Higher`, others also `Pick First` and `Second`. Smaller models also exhibit `null`. These patterns (1) differ across models (whether it is Pick First or Second vs Pick Higher or something else) but (2) the word identity (*red*/*white* with the same numeric information) consistently interacts with word order (*red*/*white* vs *white*/*red*) to create systematic error patterns per model.

6 Analysis: Number Token Frequency

Numbers appear at different frequencies in training data. Does frequency explain the differences in calibration? Across almost all models, we find a significant correlation between number frequency and average calibration scores. However, this is only a correlation, and further study is required to causally understand model behavior.

(5) From **50** *blue* marbles and **84** *red*. . .

Repeating an example similar to those above, here we focus on the frequencies of the numeric content numeric, $N_1 = 50/N_2 = 84$, in the training data. (Frequencies of the options (e.g., *blue*, *red*) were statistically insignificant with little effect.) Because the models we test do not publish their training data details, we approximate the frequency of numbers in their training data using infini-gram[6] (Liu et al., 2024) over the 3 trillion token corpus Dolma (Soldaini et al., 2024). Given the limited amount of natural text available online, we expect pre-training datasets to overlap and be similarly distributed.[7]

We use the Spearman's correlation between the calibration scores and the *Frequency Gap*, the log absolute difference of the number frequencies, $\log|(\mathrm{Freq}(N_1) - \mathrm{Freq}(N_2)|$.[8] Using the Frequency

[6] `https://infini-gram.io/api_doc.html`

[7] We query for the number counts with spaces both before and after each number: "_N_", to account for numbers appearing within other numbers assuming the introduced noise is distributed similarly across numbers.

[8] See Appendix I for more results and correlations.

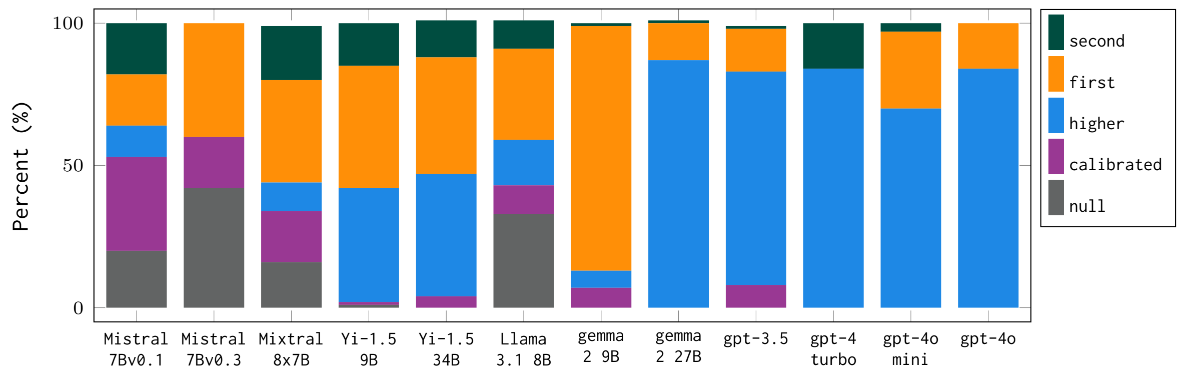

Figure 3: **Systematic Patterns in Model Behavior** for the *colors* dataset. Each bar shows the percent of the different behaviors models exhibit averaged across templates, color pairs, and numeric scales.

	Base			Chat		
	colors	*wordp.*	*distr.*	*colors*	*wordp.*	*distr.*
Llama-3.1-8B	+0.68	+0.46	-0.26	+0.57	+0.03	-0.09
Mistral-7B-v0.1	+0.58	+0.60	-0.40	+0.17	+0.08	-0.25
Mistral-7B-v0.3	+0.57	+0.59	-0.44	+0.21	+0.00	+0.02
Mixtral-8x7B	+0.71	+0.52	-0.34	-0.44	-0.04	+0.00
Yi-1.5-34B	+0.66	+0.49	-0.35	-0.01	+0.02	-0.00
Yi-1.5-9B	+0.57	+0.49	-0.48	-0.24	-0.06	-0.20
gemma-2-27b	+0.61	+0.52	-0.44	-0.55	-0.57	-0.12
gemma-2-9b	+0.55	+0.65	-0.47	-0.08	-0.02	-0.16
gpt-3.5	-	-	-	-0.40	-0.38	-0.12
gpt-4-turbo	-	-	-	-0.54	-0.47	-0.01
gpt-4o	-	-	-	-0.46	-0.34	-0.24
gpt-4o-mini	-	-	-	-0.61	-0.07	-0.31

Table 6: **Frequency Effects.** Each cell reports the Spearman's correlation between the *Frequency Gap* between the numbers in context and the calibration scores. Positive correlations mean that larger word frequency gaps predict worse calibration, highlighted orange (worse). Negative correlations mean that larger frequency gaps predict improved calibration, highlighted blue (better). Cells not highlighted are not statistically significant at $p < 0.01$.

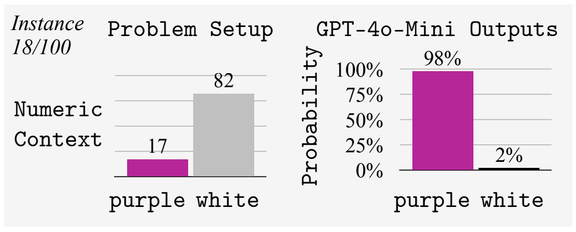

Figure 4: **Representative Examples of Model Behaviors.** This instance is from `gpt-4o-mini` results in Figure 2, `Pick First` behavior from the top-left cell.

Gap is a simple way to capture if there is a relationship between the frequencies of the two numbers.

We present the correlation results in Table 6. Frequency effects appear to negatively impact the base models: Larger Frequency Gaps correlate with worse calibration. Conversely, frequency effects appear to positively impact instruction-tuned models: Larger Frequency Gaps correlate with improved calibration. Frequency effects on *distributions* differ; Effects are weaker, and increased frequency improves calibration. Frequency effects are a known issue for LMs (Wei et al., 2021; Lovering and Pavlick, 2023; McCoy et al., 2024); **Our results suggest that increases in model scale and performance have not canceled out such biases.** As mentioned above, this study is not causal, and while we hope it is helpful, only is a correlative result. Controlled methods that involve re-training models would better yield a more complete understanding of how frequency and calibration relate.

7 Conclusion

All models we test are generally less calibrated than simple baselines. Word identity, word order, and the vocabulary frequency of the numeric information correlate with and appear to modulate model behavior. Even in the basic problem settings we test in this paper, LMs are biased, suggesting there are significant risks to using models in probabilistic scenarios to make decisions.

Limitations

This work strikes a balance between sampling across a large space of options–more numbers (Shrestha et al., 2025), more unique templates–and uncovering biases present in those chosen options, which benefits from a "grid-search" approach. Future work that increases the diversity and size of these datasets could only improve this line of work.

We report Spearman's correlation in Section 6. Using mixed-effects models may provide additional nuance. Increasing the range of numbers used, especially in this section, could improve the robustness of our results.

Potential Risks

This work does not pose a risk. Instead, this work highlights several risks of using uncalibrated models and shows that current models are uncalibrated even in simple settings.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. arXiv preprint arXiv:2204.05862.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D 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 Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In Advances in Neural Information Processing Systems, volume 33, pages 1877–1901. Curran Associates, Inc.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374.

Yuri Chervonyi, Trieu H. Trinh, Miroslav Olšák, Xiaomeng Yang, Hoang Nguyen, Marcelo Menegali, Junehyuk Jung, Vikas Verma, Quoc V. Le, and Thang Luong. 2025. Gold-medalist performance in solving olympiad geometry with alphageometry2. Preprint, arXiv:2502.03544.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

Kfir Eliaz and Ariel Rubinstein. 2011. Edgar allan poe's riddle: Framing effects in repeated matching pennies games. Games and Economic Behavior, 71(1):88–99.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. 2017. On calibration of modern neural networks. In International conference on machine learning, pages 1321–1330. PMLR.

Ritwik Gupta, Rodolfo Corona, Jiaxin Ge, Eric Wang, Dan Klein, Trevor Darrell, and David M Chan. 2025. Enough coin flips can make llms act bayesian. arXiv preprint arXiv:2503.04722.

Aspen K Hopkins, Alex Renda, and Michael Carbin. 2023. Can llms generate random numbers? evaluating llm sampling in controlled domains. In ICML 2023 Workshop: Sampling and Optimization in Discrete Space.

Janus. 2022. Mysteries of mode collapse. https://www.lesswrong.com/posts/t9svvNPNmFf5Qa3TA/mysteries-of-mode-collapse. Accessed: 2024-09-01.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, and 1 others. 2023. Mistral 7b. arXiv preprint arXiv:2310.06825.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. arXiv preprint arXiv:2401.04088.

Leonid V Kantorovich. 1939. The mathematical method of production planning and organization. Management Science, 6(4):363–422.

Abhishek Kumar, Robert Morabito, Sanzhar Umbet, Jad Kabbara, and Ali Emami. 2024. Confidence under the hood: An investigation into the confidence-probability alignment in large language models. arXiv preprint arXiv:2405.16282.

Matt J Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual fairness. Advances in neural information processing systems, 30.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Teaching models to express their uncertainty in words. Transactions on Machine Learning Research.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and 1 others. 2024. Rho-1: Not all tokens are what you need. arXiv preprint arXiv:2404.07965.

Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. arXiv preprint arXiv:2401.17377.

Charles Lovering and Ellie Pavlick. 2023. Training priors predict text-to-image model performance. arXiv preprint arXiv:2306.01755.

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

Nicole Meister, Carlos Guestrin, and Tatsunori Hashimoto. 2024. Benchmarking distributional alignment of large language models. arXiv preprint arXiv:2411.05403.

Sabrina J Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. 2022. Reducing conversational agents' overconfidence through linguistic calibration. Transactions of the Association for Computational Linguistics, 10:857–872.

Evan Miller. 2024. Adding error bars to evals: A statistical approach to language model evaluations. arXiv preprint arXiv:2411.00640.

Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Ann Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. 2021. Revisiting the calibration of modern neural networks. In Advances in Neural Information Processing Systems.

MistralAI. 2025. Mathstral.

OpenAI. 2024. Gpt-4o system card. `https://openai.com/index/gpt-4o-system-card/`. Accessed: 2024-09-01.

Laura O'Mahony, Leo Grinsztajn, Hailey Schoelkopf, and Stella Biderman. 2024. Attributing mode collapse in the fine-tuning of large language models. ICLR 2024 Workshop on Understanding of Foundation Models.

Pouya Pezeshkpour and Estevam Hruschka. 2024. Large language models sensitivity to the order of options in multiple-choice questions. In Findings of the Association for Computational Linguistics: NAACL 2024, pages 2006–2017, Mexico City, Mexico. Association for Computational Linguistics.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, and 1 others. 2025. Humanity's last exam. arXiv preprint arXiv:2501.14249.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, Louisiana. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300.

Advik Shreekumar. 2025. X-raying experts: Decomposing predictable mistakes in radiology.

Safal Shrestha, Minwu Kim, and Keith Ross. 2025. Mathematical reasoning in large language models: Assessing logical and arithmetic errors across wide numerical ranges. Preprint, arXiv:2502.08680.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, and 1 others. 2024. Dolma: An open corpus of three trillion tokens for language model pretraining research. arXiv preprint arXiv:2402.00159.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. arXiv preprint arXiv:2408.00118.

Polina Tsvilodub, Kanishk Gandhi, Haoran Zhao, Jan-Philipp Fränken, Michael Franke, and Noah D Goodman. 2025. Non-literal understanding of number words by language models. arXiv preprint arXiv:2502.06204.

Katherine Van Koevering and Jon Kleinberg. 2024. How random is random? evaluating the randomness and humaness of llms' coin flips. arXiv preprint arXiv:2406.00092.

A. Wald and J. Wolfowitz. 1940. On a test whether two samples are from the same population. The Annals of Mathematical Statistics, 11(2).

Jason Wei, Dan Garrette, Tal Linzen, and Ellie Pavlick. 2021. Frequency effects on syntactic rule learning in transformers. arXiv preprint arXiv:2109.07020.

Jason Wei, Nguyen Karina, Hyung Won Chung, Yunxin Joy Jiao, Spencer Papay, Amelia Glaese, John Schulman, and William Fedus. 2024. Measuring short-form factuality in large language models. arXiv preprint arXiv:2411.04368.

Gal Yona, Roee Aharoni, and Mor Geva. 2024. Can large language models faithfully express their intrinsic uncertainty in words? arXiv preprint arXiv:2405.16908.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, and 1 others. 2024. Yi: Open foundation models by 01. ai. arXiv preprint arXiv:2403.04652.

Kaitlyn Zhou, Jena Hwang, Xiang Ren, and Maarten Sap. 2024. Relying on the unreliable: The impact of language models' reluctance to express uncertainty. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3623–3643, Bangkok, Thailand. Association for Computational Linguistics.

Kaitlyn Zhou, Dan Jurafsky, and Tatsunori Hashimoto. 2023. Navigating the grey area: How expressions of uncertainty and overconfidence affect language models. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 5506–5524, Singapore. Association for Computational Linguistics.

Basics

A Housekeeping

- First off, we apologize to any color-blind readers; we make heavy use of color in this paper. We do our best to use color-blind aware color palettes. (We found David Nichols's resources helpful: https://davidmathlogic.com/colorblind/#%23D81B60-%231E88E5-%23FFC107-%23004D40.)
- We used ChatGPT to assist with LaTeX debugging. (Editorial comment: it was net helpful if not always right.)
- Because we have a large number of figures in this appendix, we front-load the (almost) text-only sections first and then follow this with a single column, mirrored section structure, with only figures.

B Reproducibility

All data and experiments will be released under The MIT License. All details required for reproducing our results, like problem templates and prompts, will also be released at the link. Data was created with research settings in mind, but can be used in any setting (research/professional). https://drive.google.com/drive/folders/11mY5vbNanqMA2wwci84I5cipwh7FPPPB

Appendix K report the prompts (formatted as messages) and problem templates.

We use the models listed in Table 9. For visual acuity, we only put the full model codes in this table, rather than throughout the paper. All experiments can be run on a P4de-24 (8 NVIDIA A100) in approximately 48 hours.

C Additional Motivation

The training objective of language modeling will capture whatever pattern of calibration best describes data in the wild, not necessarily probability or rank. If natural language texts show examples that are calibrated then models should learn to be calibrated. If natural language texts show examples that are anti-calibrated then models might even learn an inverted rank-order. That would be a strange result, but it is conceivable. (Perhaps, if models were only trained on fictional stories where only unlikely events occur, then we could see this type of result.) However, if data and events described by language model training data were distributed naturally then minimizing the cross-entropy would yield a calibrated model.

Take for example a token that is purported to occur with 80% probability. If we sum the loss across 100 tokens where this token occurs 80 times and the alternate occurs 20 we can see that the losses for a calibrated model are lower than one that only gets the rank-order correct placing almost 100% of the probability mass on the correct token every time.

$$\begin{aligned} loss(\text{rank-order}) &= -log(1.00 - \epsilon) \times 80 + -log(\epsilon) \times 20 \\ &= -log(0.99) \times 80 + -log(0.01) \times 20 \\ &= 0.004 \times 80 + 2 \times 20 \\ &= 40.349 \\ loss(\text{calibrated}) &= -log(0.8) \times 80 + -log(0.2) \times 20 \\ &= 0.10 \times 80 + 0.70 \times 20 \\ &= 21.73 \end{aligned}$$

This is true generally; the loss for the rank-order model will increase with decreasing epsilon. Assuming that the training dataset is calibrated to some p you can show that the optimal probabilities for a model to learn is the calibrated probability p, by solving for the derivative of the loss with respect to the output probability in question and setting it to zero. There is still, of course, the empirical question of how the training data is distributed and what kind of behavior we would want from our models.

More Results

D More Results on What Do Metric Values Mean?

In this section, we show reference performance scores, Table 10. Second, we show that most models are capable of doing well and sometimes (albeit rarely) do in fact produce well-calibrated outputs. The results here are over a small, new set of numbers for demonstration purposes: 8, 9, 48, 51, 98, 99. Figures are in Appendix L.

The scores in Table 10 show WD on the high end will be around 1.00. What these results don't show well is what is a "good" calibration score. We find that in a small number of cases the model achieves calibration scores near 0, less than 0.05. We consider these examples solved.

E More Results on Is It Possible For The Models To Do Well?

Figures 10 to 12 shows the count of solved inputs across either pairs of numbers (left) or pairs of colors (right). Each number is a raw count out of 500, shown as such for visual acuity. For these plots we considered the model to have solved an instance if the WD < 0.05. This tends to constitute a good fit. Making this threshold more strict, say, 0.01, maintains the trends present in the figure. Mostly, these plots show that (1) in some cases the models do calibrate well even if in general they tend not to; (2) the number pair and option pair have a large impact on whether a model ever yields a solved result, mirroring previous results; (3) the patterns differ per model but again mirror previous results. The most important result is the first. **Models are clearly capable of yielding calibrated probabilties, but do not do so in the (large) majority of cases.**

F More Results on Calibration And Low Probability Mass

In this section the first question we ask is if our calibration results are stable across a greater diversity of data. We do this by 1) introducing variations of each template, and 2) creating more templates. Specifically, using `gpt-4o-2024-08-06`, we create 500 leading introductions to prefix our templates in the *colors* dataset, ranging from one to ten sentences. Next, we create 500 more templates for the *colors* dataset using `gpt-4o-2024-08-06`. For both new splits we test a subset of our instruction-tuned models. The calibration scores did changed little from the original performances, Table 7. In two cases, the calibration was worse (the confidence interval was strictly positive), though the effect was not large. Using synthetically generated data has its risks, nonetheless, we take these results as evidence that calibration over a wider distribution of similar data remains poor.

	WD	*Intros* - WD Δ (95% CI)	*Templates* - WD Δ (95% CI)
Llama-3.1-8B	0.40	(-0.04, +0.00)	(+0.02, +0.05)
Yi-1.5-9B	0.49	(-0.03, +0.02)	(+0.09,+ 0.13)
gemma-2-9b	0.50	(-0.02, +0.04)	(-0.00, +0.04)

Table 7: Calibration Scores for Additional Experiments. The first column WD is the original calibration score (repeated from Table 4). The remaining columns show the 95% confidence intervals of the change in performance between the original results and the alternate settings, *intros* and *templates*. Statistically significant increases in calibration scores are highlighted orange (worse calibration).

G More Results on Mode Collapse

Table 12 shows the drop in RE from base models to chat models. This occurs across all datasets.

We also provide a more in-depth analysis on mode collapse in the *distributions* dataset.

Additional Metrics for *distributions*. We introduce three metrics to understand if there are patterns in the mode.

Mode Probability is the mean max probability of Π over the dataset, $\text{AVG}_{\text{dataset}} \max \Pi$.

Mode Stability is the rate at which the most likely token is preserved between base and chat versions of a model averaged over the dataset, $\text{AVG}_{\text{dataset}}(\mathbb{1}(\arg\max \Pi_{\text{base}} = \arg\max \Pi_{\text{chat}}))$.

Mode Frequency examines whether there are biases for particular tokens (numbers). This metric measures the frequency of the mode averaged across distributions. Because different distributions cover different digits, we group the distributions by underlying problem range and inclusivity: E.g., [2, 5) and [132, 135) are grouped. This is equivalent to: (1) setting the temperature to 0 (greedy sampling), (2) averaging over the groups, keeping the digits dimension, (3) taking the maximum for each group, and (4) averaging across all maximums, $\text{AVG}(\max_{\text{digits}} \text{AVG}_{\text{distributions, keepdim}}(\Pi^{\tau=0}))$.

Results We find evidence that the reduced performance of chat models results from their tendency to over-allocate probability on a subset of valid tokens. Table 13 provides additional metrics that suggest mode-collapse. For chat models, the most likely token receives, on average, 66% of the probability mass (up from 28% for base models, beyond the empirical ideal 23%, on this dataset). The most likely token remains the same for 48% instances across base and chat models, suggesting that the mode is often preserved. Figure 5 reports the mode frequency for `gpt-4o-mini`. (For all models see Figure 13.) If all probability mass were on valid tokens, the minimum mode frequency would be the same as the rate of the uniform distribution; we see much higher rates, ranging from $+29$ to $+50\%$. The implication is that models are also uncalibrated at an outcome level with systematic biases (preferences) for certain numbers.

H More Results on Ordering and Colors

We categorize model output behavior over sets of results by using computing the distance to

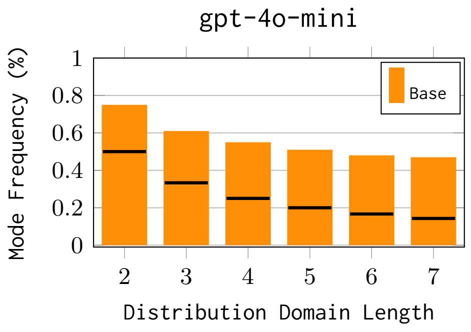

Figure 5: **Models Over-represent Some Numbers**; the modes are heavily over-represented. Each bar is the mode frequency, i.e., how often the top-chosen token is chosen averaged over distributions. The black lines mark the expected rate for a calibrated model.

reference behaviors. Figure 3 computes the proportion of each behavior by classifying a set of results (across an number scale, template, and option pair) as one of the reference points. This is done by measuring the WD between the output probabilities and reference point. We additionally classify any grid where the average PM is less than 0.5 as Null. Meaning, anything where more than 50% of the probability mass is on tokens outside of T are considered out of scope. This is how the results in Figure 21, Figure 20, and Figure 19 are computed.

The bias across for individual option words varies model to model, but there is some shared ordering. Figure 14 shows how win rates change when a color is placed before vs after other colors. If the color did not matter we would expect the win rates to be around 50% (with more variance). Instead, there are clear rank orders. Appendix P shows the average change in calibration (lower is better, or a reduction in calibration error).

When looking at the hierarchies formed by the options on the X-axes of Figure 14 there appears to be some shared structure across models. For example, for most models, listing the color white first leads to an improvement in calibration. Notably, this does not mean the models prefer this color or pick it. Always picking it would not be particularly calibrated. The example we saw in the main body of the paper was the `gpt-4o-mini` was biased against picking white when it was second and tended to Pick Higher when it was first.

Table 14 shows the correlation of the hierarchy orderings across models. The structure similarity (and dissimilarity) is sometimes far beyond random. Similar models generally have more correlated orderings. `gpt-4-turbo` is the most dissimilar from others.

We show a pictorial/imagistic view of all the data via collections of heatmaps. The sets of heatmaps are computed in two different ways. Figure 18 shows a *space-station-view* of all the hierarchical, diagonal patterns; Figure 16 is the legend showing what the colors correspond to. The 15 heatmaps per model are the variations across different templates and number scales. The number scales appear to have relatively little impact on the patterns. The templates, for some models, have a larger impact. The behaviors are computed as in the main body of the paper; only among the references, we report the highest rate of behavior compatible with the model's outputs.

Figure 21 shows the same results but we create the heatmaps using a different, more high-fidelity approach. We also provide a more zoomed in look at Figure 20. To assign a behavior per cell we now use the distance to each type of behavior. The cells now show distance instead of rate. (Lower is better). This is reflected by the updated legend, Figure 19.

I More Results on Frequency Effects

Here we present more results on the frequency effects and consider alternate independent variables. Firstly, we designed our experiments, in the main, to highlight how the word identity and order impacted calibration. Experiments that examine a wider range of numbers and therefore frequencies would better study this problem. Nonetheless, we find clear and strong effects and report them here.

Figure 22 shows the frequency statistics of the number tokens used in our different datasets. Figure 23 shows how those frequency stats relate to the calibration scores for `gpt-4o`. Not shown here is there is also an order effect (which number is more frequent).

J Human Study: Model vs Human Randomness

Our experiments demonstrated that language models are poorly calibrated. Here, we provide further context by comparing the proprietary model and human behavior, asking if models are as random (or not) as people. Previous work suggests that both would be *not* random (see our Related Work section). We test this again. To get at this question, we use a variation of the two-player game Matching Pennies. We focus on a variation of the game similar to Eliaz and Rubinstein (2011). Each round, the "guesser" aims to match her chosen coin face with "misleader's" choice, earning a +1 reward if their coin faces match, and -1 otherwise. The Nash Equilibrium is for both agents to act randomly. A calibrated model can trivially achieve random behavior.

J.1 Experimental Design

This experiment is based on a simple heads/tails game. Our motivation for including this experiment is to allow a human baseline in a setting where random behavior is optimal. This section makes no distinction between calibrated probabilities and unbiased outputs because we don't have access to this information for humans. However, to "perform" well, or appear calibrated, requires either the model/person to (1) produce calibrated probabilities which when sampled from will yield random outputs, (2) use an internal random process to produce uncalibrated probabilities that yield random outputs, or (3) have a systematic process for producing outputs that *appear* random enough. (Previous work would suggest that humans are limited in this regard and will likely not be random (and therefore not calibrated).) This leaves us with a one-sided problem. If the humans/models were proficient at producing random outcomes we would have to adjudicate between the above possibilities.

In a single-round game, the Nash Equilibrium is to select a coin face randomly. We set there to be 100 rounds of play. To focus on calibration, we frame the tested model as the "misleader." The misleader is prompted to submit answers, and the "guesser" is set to predict the misleader's answers. The misleader submits $m_1, m_2, \ldots, m_{100}$ answers, and the guesser bases each of its g_i answers incrementally based upon previous misleader's answers, $\{m_j \mid 0 < j < i\}$. We use a n-gram[9] based strategy as our guesser.

Models We focus on gpt-* models sampling temperature $\tau = 1$. Human behavior is drawn from a sample of 44 games of 100 rounds from 10 different volunteers with technical backgrounds up to 10 different games each.

Figure 24 reports our the implementation for the n-gram model. We find that most n from 1 to 5 do similarly well. In the paper we report a 4-gram model.

Additional Metrics Whereas previous metrics rely on model probabilities, this experiment assumes access only to outputs, meaning we cannot use the metrics used in previous sections.

Player Win Rate (PWR) is the rate the misleader wins, averaging over 100 games each for up to 100 rounds. The higher this rate, the more random the outputs of the misleader, and thus the more calibrated it is at an outcome level.

Randomness Testing (WW). Wald–Wolfowitz (Wald and Wolfowitz, 1940) tests if the elements of a sequence are mutually independent. Using this test at a significance level of 0.05, we report the proportion of samples that cannot statistically be rejected as non-random, labeling this value as **WW**. For reference, as shown in Table 8 when we sampled random sequences using `python`, 98% of the sequences could not be rejected; only 7% of human-generated samples could not be rejected.

J.2 Results

	PWR↑	WW↑
Random	0.50	0.98
Human	0.33	0.07
gpt-3.5	**0.40**	**0.43**
gpt-4-turbo	0.33	0.25
gpt-4o	0.36	0.19

Table 8: **Human Study Results: Player win rates (PWR) and statistical tests for randomness (WW).** Higher values for both metrics suggest that models are more able to produce random (and thus more calibrated) outputs. **Takeaways: (1)** `gpt-3.5` stands out as the most random model; **(2)** Models pass a greater number of the statistical tests for randomness than humans, though are still far from random; **(3)** Model sequences are predictable at rates similar to humans.

Both best-performing proprietary models and human behavior are not random. Models are prompted with the game rules and we explicitly instruct the models (and humans) to be random. Not doing so reduces model performance to near 0. PWR for models and humans are similar, WW for models is higher, but still low.

Table 8 reports the proportion of samples (where each sample is 100 coin flips) the result could not be considered non-random ($p = 0.05$): Higher WW values suggest the model is more random.

[9]Motivated by the strategy here: https://www.expunctis.com/2019/03/07/Not-so-random.html.

More model- than human-generated samples were considered random: $19 - 43\%$ vs. 7%.

Continued Figures and Tables

K Figures: Reproducibility

Our Name	HF Model Key (Base)	HF Model Key (Chat)
Yi-1.5-9B	01-ai/Yi-1.5-9B	01-ai/Yi-1.5-9B-Chat
Yi-1.5-34B	01-ai/Yi-1.5-34B	01-ai/Yi-1.5-34B-Chat
Llama-3.1-8B	meta-llama/Llama-3.1-8B	meta-llama/Llama-3.1-8B-Instruct
Mistral-7B-v0.1	mistralai/Mistral-7B-v0.1	mistralai/Mistral-7B-Instruct-v0.1
Mistral-7B-v0.3	mistralai/Mistral-7B-v0.3	mistralai/Mistral-7B-Instruct-v0.3
Mixtral-8x7B-v0.1	mistralai/Mixtral-8x7B-v0.1	mistralai/Mixtral-8x7B-Instruct-v0.1
gemma-2-9b	google/gemma-2-9b	google/gemma-2-9b-it
gemma-2-27b	google/gemma-2-27b	google/gemma-2-27b-it
		OpenAI Model Key
gpt-3.5		gpt-3.5-turbo-0125
gpt-4-turbo		gpt-4-turbo-2024-04-09
gpt-4o		gpt-4o-2024-08-06
gpt-4o-mini		gpt-4o-mini-2024-07-18

Table 9: **Models**

```
# prompt 0
[
 {
 "role": "user",
 "content": "You are a perfectly calibrated language model that produces tokens with probabilities
     ↪  that correspond to values in the context and prompt. I will give you a scenario. Produce
     ↪ only a single word and nothing else. The word will be one of the options in context.",
 },
 {
 "role": "assistant",
 "content": "Excellent. I will continue by producing a single word after your next message.",
 },
 {"role": "user", "content": text},]
# prompt 1
[
 {
 "role": "user",
 "content": "You are a perfectly calibrated language model that produces tokens with probabilities
     ↪  that correspond to values in the context and prompt. I will give you a context where
     ↪ there are some probabilities involved. Produce one token.",
 },
 {
 "role": "assistant",
 "content": "Excellent. I will generate a single token calibrated to the statistics of the context
     ↪ .",
 },
 {"role": "user", "content": text},]
# prompt 2
[
 {
 "role": "user",
 "content": "You are a master of statistics and math. Produce tokens perfectly calibrated to the
     ↪ numbers in context. Produce one token.",
 },
 {"role": "assistant", "content": "Excellent. Ask away!"},
 {"role": "user", "content": text},]
# prompt 3
[
 {
 "role": "user",
 "content": "Produce tokens with probability calibrated to the numbers in context. Produce a
     ↪ single token.",
 },
 {"role": "assistant", "content": "Excellent. Ask away!"},
 {"role": "user", "content": text},]
# prompt 4
[
 {
 "role": "user",
 "content": "Produce tokens with probability calibrated to the statistics implicit and explicit in
     ↪  context. Produce only a single token.",
 },
 {"role": "assistant", "content": "Excellent. Ask away!"},
 {"role": "user", "content": text}, ],
```

We use the message format shared by OpenAI and HuggingFace. The problem {text} is injected into the prompt message.

Figure 6: **Prompts For Instruction-Tuned Models; `Colors/Wordproblems`.**

```
# prompt 0
[
 {
 "role": "user",
 "content": "You are a perfectly calibrated language model that produces tokens with probabilities
     ↪  that correspond to values in the context and prompt. I will give you a sentence. Finish
     ↪ the sentence by producing only a number and nothing else.",
 },
 {
 "role": "assistant",
 "content": "Excellent. I will continue by producing a single number after your next message.",
 },
 {"role": "user", "content": @text},
]
# prompt 1
[
 {
 "role": "user",
 "content": "You are a master of statistics and math. Produce tokens perfectly calibrated to the
     ↪ numbers in context.  I will give you a sentence. Produce one number to finish the sentence.
     ↪ ",
 },
 {"role": "assistant", "content": "Excellent. Ask away!"},
 {"role": "user", "content": @text},
]
# prompt 2
[
 {
 "role": "user",
 "content": "Produce tokens with probability calibrated to the numbers in context. Given the
     ↪ scenario, produce a number.",
 },
 {"role": "assistant", "content": "Excellent. Go ahead!"},
 {"role": "user", "content": @text},
]
```

We use the message format shared by OpenAI and HuggingFace. The problem {text} is injected into the prompt message.

Figure 7: **Prompts For Instruction-Tuned Models; `Distributions`.**

```
# template 0
"There were {number_1} {option_1} marbles and {number_2} {option_2} marbles in a bag. Jane
     ↪ randomly picked a marble and saw it was the color"
# template 1
"Billy got to pick one marble from a big urn with many marbles. There were {number_1} {option_1}
     ↪ marbles and {number_2} {option_2} marbles in an urn. The color of the marble Billy
     ↪ randomly picked was"
# template 2
"Amanda had a huge pile of shirts. There were {number_1} {option_1} shirts and {number_2} {
     ↪ option_2} shirts. Without looking, she picked one by chance. The color of the shirt was"
# template 3
"Bill and Rick went to the hardwore store for paint in a hurry. The store had {number_1} shades of
     ↪  {option_1} and {number_2} shades of {option_2}. They didn't have any time to test out
     ↪ colors so they randomly grabbed a can. The color they grabbed turned out to be"
# template 4
"Kids at soccer practice randomly grabbed pinnies from a bag. There were {number_1} {option_1}
     ↪ pinnies and {number_2} {option_2} pinnies. Tommy's pinny was the color"
```

The variables are injected into the text; {number_1}, {number_2}, {option_1}, {option_2}.

Figure 8: **Templates for `Probabilities`.**

```
# template 0
"Picking an integer between {number_1} inclusive and {number_2} exclusive, the integer I got was{
    ↪ prefix}"
# template 1
"Picking an integer between {number_1} and {number_2}, exclusive on both sides, the integer I got
    ↪ was{prefix}"
# template 2
"I sampled an integer from a random variable distributed uniformly between {number_1} inclusive
    ↪ and {number_2} exclusive, getting the value{prefix}"
# template 3
"I sampled an integer from a random variable distributed uniformly between {number_1} and {
    ↪ number_2}, exclusive on both ends, getting the value{prefix}"
```

The variables are injected into the text; {number_1}, {number_2}, {prefix}. {prefix} is always set to empty; early pilot experiments suggest that using an ellipsis leads to a similar pattern of results.

Figure 9: **Templates for `Distributions`.**

L Figures: What Do Metric Values Mean?

Settings: `51` `purple`, `98` `white`					
	$P(t_1)$	$P(t_2)$	PM	RE	WD
Ideal	0.342	0.657	1.000	1.000	0.000
Pick First	1.000	0.000	1.000	-0.927	0.930
Pick Second	0.000	1.000	1.000	-0.927	0.484
Pick Higher$_{p=0.7}$	0.300	0.700	1.000	-0.046	0.060
Pick Higher$_{p=0.6}$	0.400	0.600	1.000	0.044	0.082
Pick Higher$_{p=0.8}$	0.200	0.800	1.000	-0.205	0.201
Pick Higher$_{p=0.9}$	0.100	0.900	1.000	-0.458	0.343
Pick Higher	0.000	1.000	1.000	-0.927	0.484
Pick Lower	1.000	0.000	1.000	-0.927	0.930
Meta-Llama-3.1-8B	0.687	0.279	0.966	-0.060	0.512
Mistral-7B-v0.1	0.698	0.024	0.723	-0.715	0.727
Mistral-7B-v0.3	0.938	0.000	0.938	-0.926	0.887
Mixtral-8x7B-v0.1	1.000	0.000	1.000	-0.927	0.930
Yi-1.5-34B	0.581	0.010	0.591	-0.800	0.690
Yi-1.5-9B	0.999	0.000	0.999	-0.924	0.929
gpt-4-turbo	0.014	0.985	1.000	-0.818	0.464
gpt-4o	0.245	0.755	1.000	-0.124	0.137
gpt-4o-mini	0.998	0.002	1.000	-0.902	0.927

Settings: `51` `white`, `98` `purple`					
	$P(t_1)$	$P(t_2)$	PM	RE	WD
Ideal	0.342	0.657	1.000	1.000	0.000
Meta-Llama-3.1-8B	0.027	0.955	0.982	-0.747	0.433
Mistral-7B-v0.1	0.304	0.361	0.665	0.068	0.299
Mistral-7B-v0.3	0.035	0.886	0.921	-0.694	0.383
Mixtral-8x7B-v0.1	0.101	0.899	1.000	-0.455	0.341
Yi-1.5-34B	0.179	0.147	0.326	0.066	0.536
Yi-1.5-9B	0.951	0.048	0.999	-0.649	0.862
gpt-4-turbo	0.000	1.000	1.000	-0.927	0.484
gpt-4o	0.095	0.905	1.000	-0.473	0.349
gpt-4o-mini	0.095	0.905	1.000	-0.473	0.349

Table 10: Each row is a single result for a single setting. The first row shows the ideal, calibrated result. The first section of each tables shows baseline values, which are helpful for understanding the practical ranges of RE and WD values. The subsequent rows show how different models behave. These values, see in particular `gpt-4o-mini`, line up with the summary behaviors seen in Figure 2 and Appendix Figure 20. In the second section we invert the order of the color options. No model produces a well-calibrated result for this problem.

M Figures: More Results on Is It Possible For The Models To Do Well?

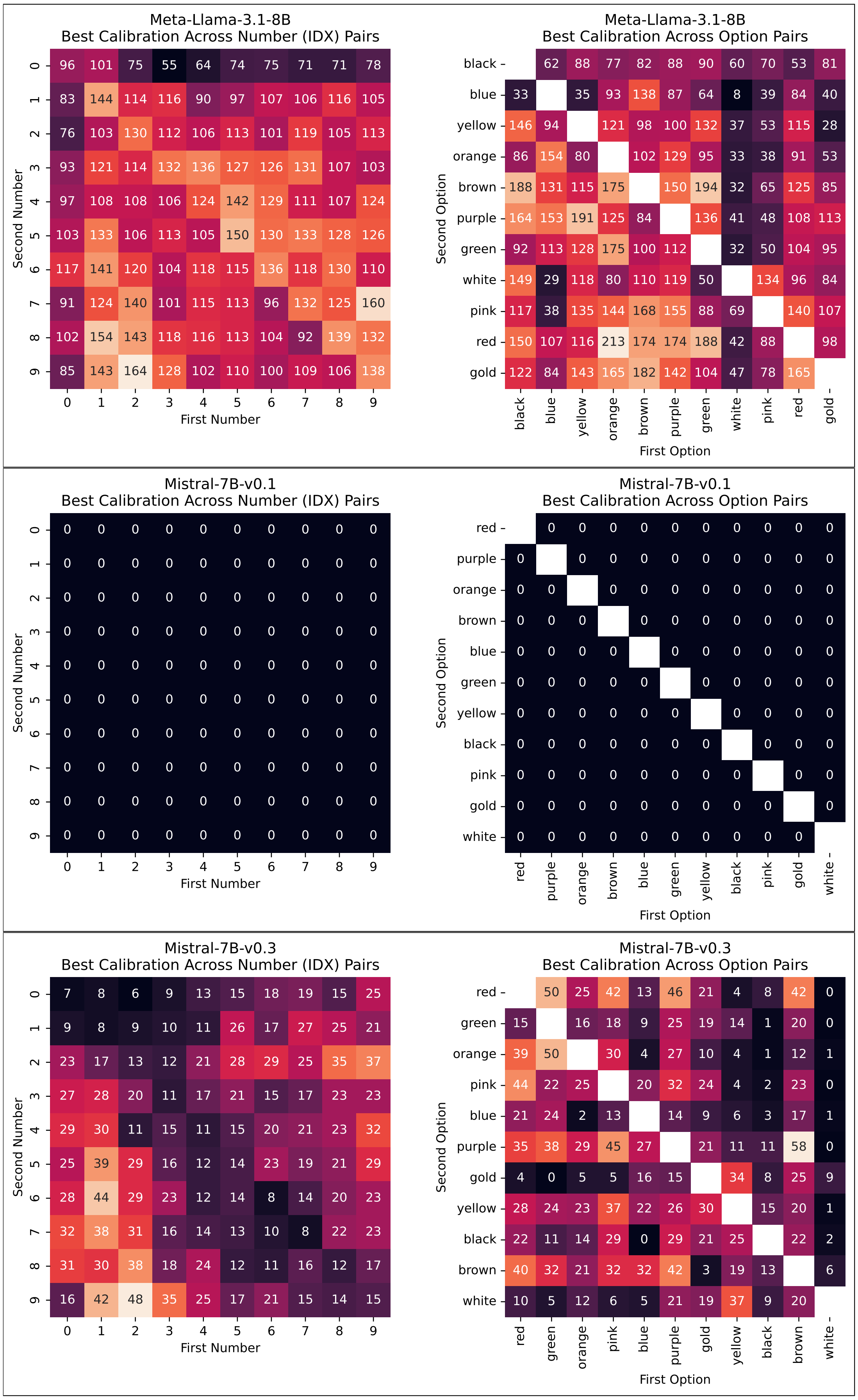

Figure 10: **Best Results Over Different Settings for `Llama-3.1-8B`, `Mistral-7B-v0.1`, `Mistral-7B-v0.3`.**

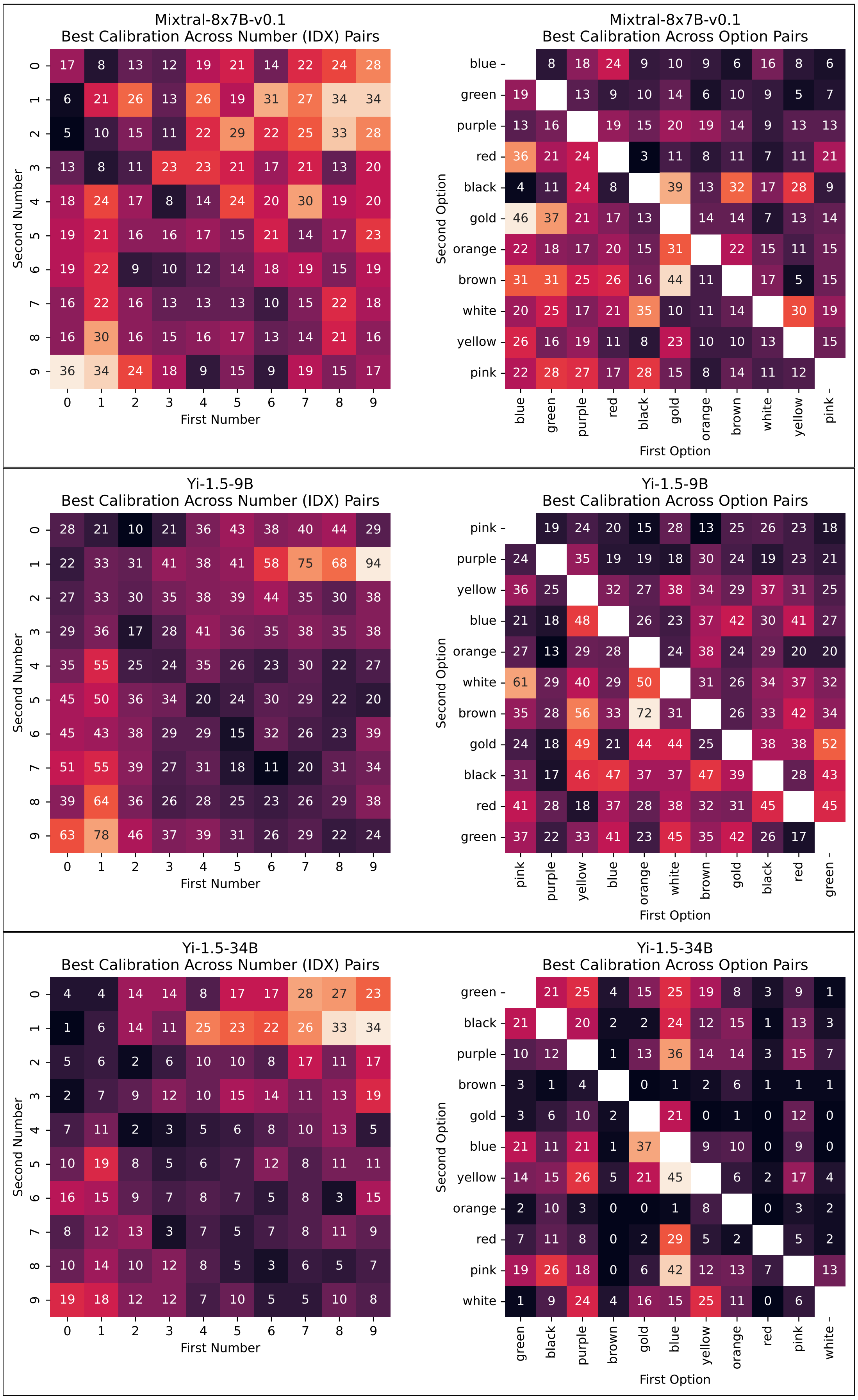

Figure 11: **Best Results Over Different Settings for Mixtral-8x7B-v0.1, `Yi-1.5-9B`, `Yi-1.5-34B`.**

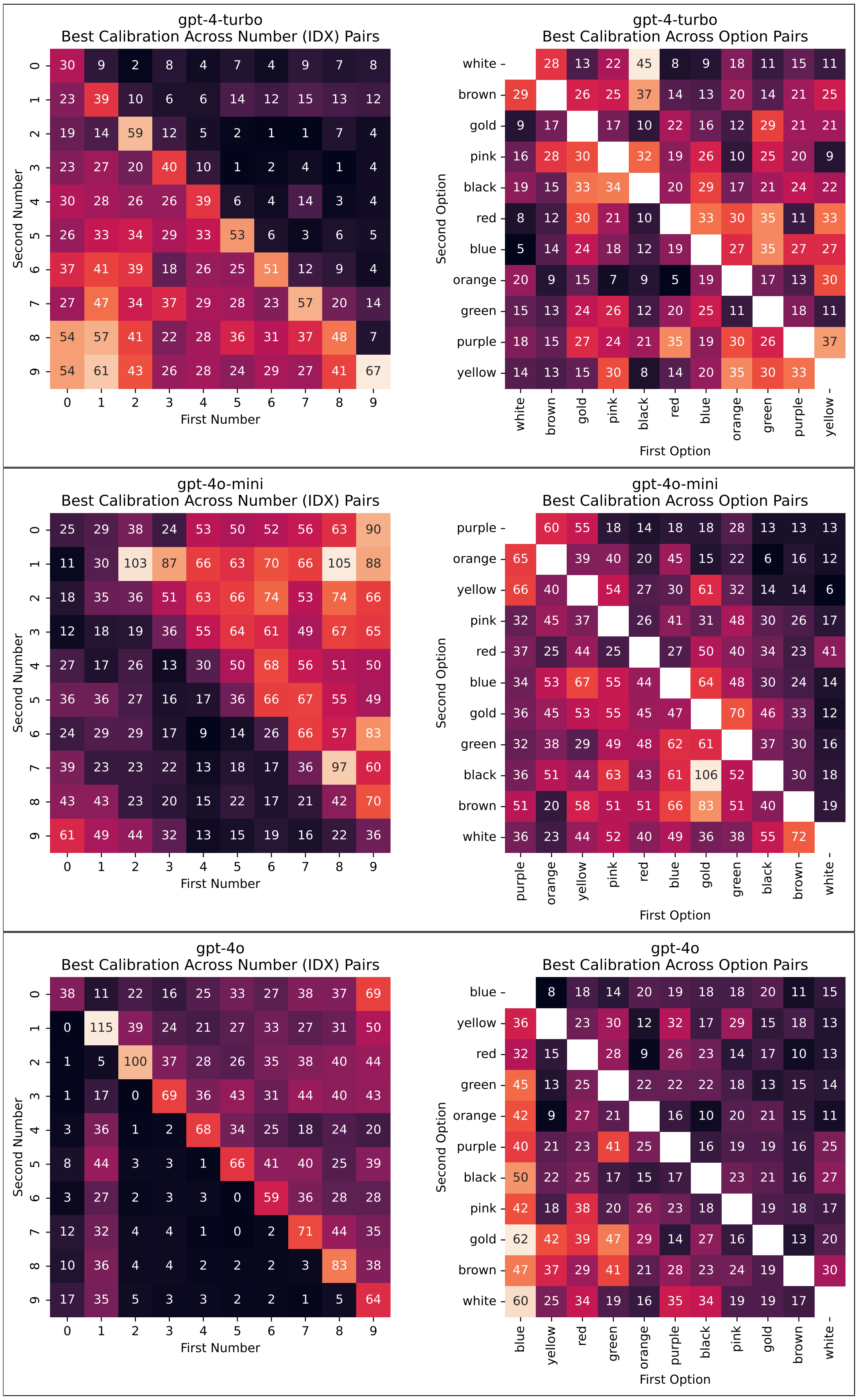

Figure 12: **Best Results Over Different Settings for `gpt-4-turbo`, `gpt-4o-mini`, `gpt-4o`.**

N Figures: More Results on Calibration And Low Probability Mass

	colors		*wordproblems*		*distributions*	
Pick Higher	$0.47_{\pm 0.01}$		$0.44_{\pm 0.00}$		-	
Pick Higher$_{p=0.7}$	$0.15_{\pm 0.01}$		$0.17_{\pm 0.00}$		-	
Pick Lower	$0.95_{\pm 0.01}$		$0.98_{\pm 0.00}$		-	
Pick First/Second	$0.71_{\pm 0.02}$		$0.71_{\pm 0.00}$		$0.38_{\pm 0.00}$	
Random$_{\tau=0.01}$	$0.71_{\pm 0.02}$		$0.69_{\pm 0.00}$		$0.86_{\pm 0.00}$	
Random$_{\tau=1.0}$	$0.27_{\pm 0.01}$		$0.29_{\pm 0.00}$		$0.38_{\pm 0.00}$	
	Base	Chat	Base	Chat	Base	Chat
Llama-3.1-8B	$0.55_{\pm 0.00}$	$0.40_{\pm 0.01}$	$0.53_{\pm 0.03}$	$0.48_{\pm 0.02}$	$0.17_{\pm 0.00}$	$0.43_{\pm 0.01}$
Mistral-7B-v0.1	$0.60_{\pm 0.00}$	$0.50_{\pm 0.01}$	$0.55_{\pm 0.03}$	$0.49_{\pm 0.01}$	$0.13_{\pm 0.00}$	$0.22_{\pm 0.01}$
Mistral-7B-v0.3	$0.58_{\pm 0.00}$	$0.48_{\pm 0.02}$	$0.57_{\pm 0.03}$	$0.55_{\pm 0.01}$	$0.13_{\pm 0.00}$	$0.56_{\pm 0.01}$
Mixtral-8x7B-v0.1	$0.56_{\pm 0.00}$	$0.51_{\pm 0.01}$	$0.51_{\pm 0.03}$	$0.60_{\pm 0.03}$	$0.16_{\pm 0.01}$	$0.71_{\pm 0.01}$
Yi-1.5-34B	$0.54_{\pm 0.01}$	$0.55_{\pm 0.01}$	$0.56_{\pm 0.04}$	$0.57_{\pm 0.01}$	$0.21_{\pm 0.01}$	$0.62_{\pm 0.01}$
Yi-1.5-9B	$0.60_{\pm 0.01}$	$0.49_{\pm 0.01}$	$0.53_{\pm 0.02}$	$0.56_{\pm 0.01}$	$0.18_{\pm 0.01}$	$0.61_{\pm 0.01}$
gemma-2-27b	$0.47_{\pm 0.00}$	$0.40_{\pm 0.00}$	$0.50_{\pm 0.04}$	$0.48_{\pm 0.01}$	$0.13_{\pm 0.01}$	$0.59_{\pm 0.01}$
gemma-2-9b	$0.54_{\pm 0.00}$	$0.50_{\pm 0.01}$	$0.48_{\pm 0.03}$	$0.52_{\pm 0.02}$	$0.14_{\pm 0.00}$	$0.59_{\pm 0.01}$
gpt-3.5	-	$0.30_{\pm 0.01}$	-	$0.57_{\pm 0.02}$	-	$0.42_{\pm 0.01}$
gpt-4-turbo	-	$0.42_{\pm 0.01}$	-	$0.62_{\pm 0.02}$	-	$0.69_{\pm 0.01}$
gpt-4o	-	$0.40_{\pm 0.00}$	-	$0.57_{\pm 0.02}$	-	$0.49_{\pm 0.01}$
gpt-4o-mini	-	$0.40_{\pm 0.01}$	-	$0.57_{\pm 0.02}$	-	$0.57_{\pm 0.01}$

Table 11: **Calibration Results.** Across two datasets, this plot shows the calibration scores (WassersteinSistance, WD) for chat versions of models along with 95% confidence intervals. (Lower Better). For *colors*, six models have a statistically better calibration than the baseline of always picking the option associated with the higher number, denoted ♠. For *wordproblems*, no model is better than this simple baseline. For *distributions*, the Base models do better, but this partially because models are have low probability mass on all the options.

O Figures: More Results on Mode Collapse

	Base	Chat	CI(Δ)
colors			
Meta-Llama-3.1-8B	$0.08_{\pm 0.01}$	$-0.11_{\pm 0.02}$	(-0.21, -0.17)
Mistral-7B-v0.1	$0.10_{\pm 0.01}$	$-0.21_{\pm 0.02}$	(-0.33, -0.29)
Mistral-7B-v0.3	$0.09_{\pm 0.01}$	$-0.36_{\pm 0.03}$	(-0.47, -0.42)
Mixtral-8x7B-v0.1	$0.06_{\pm 0.01}$	$-0.67_{\pm 0.01}$	(-0.74, -0.72)
Yi-1.5-9B	$0.06_{\pm 0.02}$	$-0.55_{\pm 0.01}$	(-0.63, -0.59)
Yi-1.5-34B	$0.02_{\pm 0.02}$	$-0.45_{\pm 0.02}$	(-0.49, -0.45)
gemma-2-9b	$0.13_{\pm 0.01}$	$-0.44_{\pm 0.01}$	(-0.58, -0.56)
gemma-2-27b	$0.06_{\pm 0.01}$	$-0.63_{\pm 0.01}$	(-0.7, -0.69)
gpt-3.5	-	$-0.35_{\pm 0.03}$	-
gpt-4-turbo	-	$-0.65_{\pm 0.01}$	-
gpt-4o-mini	-	$-0.53_{\pm 0.01}$	-
gpt-4o	-	$-0.61_{\pm 0.00}$	-
wordproblems			
Meta-Llama-3.1-8B	$-0.11_{\pm 0.04}$	$-0.33_{\pm 0.06}$	(-0.27, -0.17)
Mistral-7B-v0.1	$-0.10_{\pm 0.05}$	$-0.24_{\pm 0.04}$	(-0.18, -0.1)
Mistral-7B-v0.3	$-0.12_{\pm 0.05}$	$-0.47_{\pm 0.04}$	(-0.37, -0.33)
Mixtral-8x7B-v0.1	$-0.13_{\pm 0.07}$	$-0.65_{\pm 0.05}$	(-0.59, -0.47)
Yi-1.5-9B	$-0.11_{\pm 0.05}$	$-0.55_{\pm 0.03}$	(-0.48, -0.4)
Yi-1.5-34B	$-0.17_{\pm 0.05}$	$-0.53_{\pm 0.04}$	(-0.4, -0.32)
gemma-2-9b	$-0.07_{\pm 0.04}$	$-0.45_{\pm 0.03}$	(-0.41, -0.34)
gemma-2-27b	$-0.07_{\pm 0.06}$	$-0.57_{\pm 0.02}$	(-0.56, -0.44)
gpt-3.5	-	$-0.68_{\pm 0.04}$	-
gpt-4-turbo	-	$-0.69_{\pm 0.04}$	-
gpt-4o-mini	-	$-0.58_{\pm 0.05}$	-
gpt-4o	-	$-0.63_{\pm 0.04}$	-
distributions			
Meta-Llama-3.1-8B	$0.34_{\pm 0.02}$	$-0.70_{\pm 0.03}$	(-1.06, -1.01)
Mistral-7B-v0.1	$0.24_{\pm 0.01}$	$0.05_{\pm 0.02}$	(-0.21, -0.18)
Mistral-7B-v0.3	$0.25_{\pm 0.01}$	$-1.07_{\pm 0.04}$	(-1.36, -1.28)
Mixtral-8x7B-v0.1	$0.04_{\pm 0.01}$	$-1.59_{\pm 0.04}$	(-1.67, -1.59)
Yi-1.5-9B	$0.19_{\pm 0.01}$	$-1.13_{\pm 0.05}$	(-1.36, -1.27)
Yi-1.5-34B	$0.04_{\pm 0.01}$	$-1.32_{\pm 0.04}$	(-1.4, -1.33)
gemma-2-9b	$0.34_{\pm 0.02}$	$-1.10_{\pm 0.04}$	(-1.49, -1.39)
gemma-2-27b	$0.09_{\pm 0.01}$	$-1.14_{\pm 0.05}$	(-1.28, -1.19)
gpt-3.5	-	$-0.69_{\pm 0.02}$	-
gpt-4-turbo	-	$-1.54_{\pm 0.03}$	-
gpt-4o-mini	-	$-1.30_{\pm 0.02}$	-
gpt-4o	-	$-0.92_{\pm 0.02}$	-

Table 12: **Relative Entropy (RE) Results.** The RE is far below a calibrated level across all datasets. That being said, on all datasets, placing all probability mass on a single option (Pick First) has lower RE than models' behavior, suggesting that models have some success. This table also shows how this large drop in entropy occurred with instruction tuning. See Table 5 for the subset of results we show in the main body of the paper. **Table 5 captured differences in entropy relative to the target calibrated entropy. Here, in this table, the rightmost column captures the difference in entropy between the Base and Chat models.**

	Mode Prob Base	Mode Prob Chat	Mode Stability
Idealized	0.23	0.23	0.05
Meta-Llama-3.1-8B	0.25	0.60	0.48
Mistral-7B-v0.1	0.26	0.33	0.58
Mistral-7B-v0.3	0.26	0.65	0.35
Mixtral-8x7B-v0.1	0.32	0.82	0.52
Yi-1.5-34B	0.75	0.53	
Yi-1.5-9B	0.30	0.75	0.49
gemma-2-27b	0.29	0.68	0.46
gemma-2-9b	0.22	0.66	0.40
AVG$_{\text{OpenSource}}$	0.28	0.66	0.48
gpt-3.5	-	0.53	-
gpt-4-turbo	-	0.79	-
gpt-4o	-	0.58	-
gpt-4o-mini	-	0.73	-
AVG$_{\text{Proprietary}}$	-	0.66	-

Table 13: ***Distributions* Results.** Mode-related metrics. **(1)** Averaged, the probability mass on the top-token for chat models is 43% above the calibrated ideal, showing where relative entropy increased. **(2)** The top-tokens remain the same across base and chat models for 48% of instances. These results together suggest mode collapse.

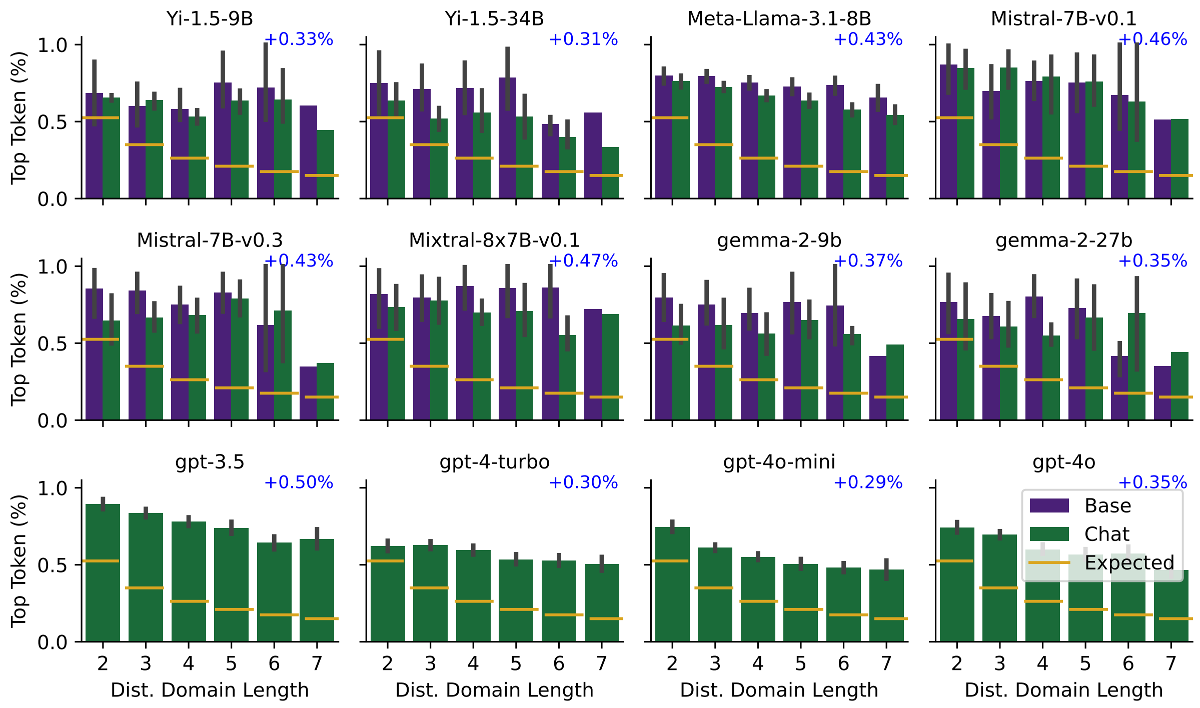

Figure 13: **Models Over-represent Numbers.** Each bar shows how often the top-chosen token is chosen (percent). The gold bars mark the expected rate for a calibrated model; the blue annotation marks the average excess probability on the top-chosen token, ranging from $29-50\%$ across models. Takeaways **(1)**: Models over-represent a token (number) over other valid options. This token is not always the same but the pattern of over-representing a number irrespective of the numeric context holds.

P Figures: More Results on Ordering and Colors

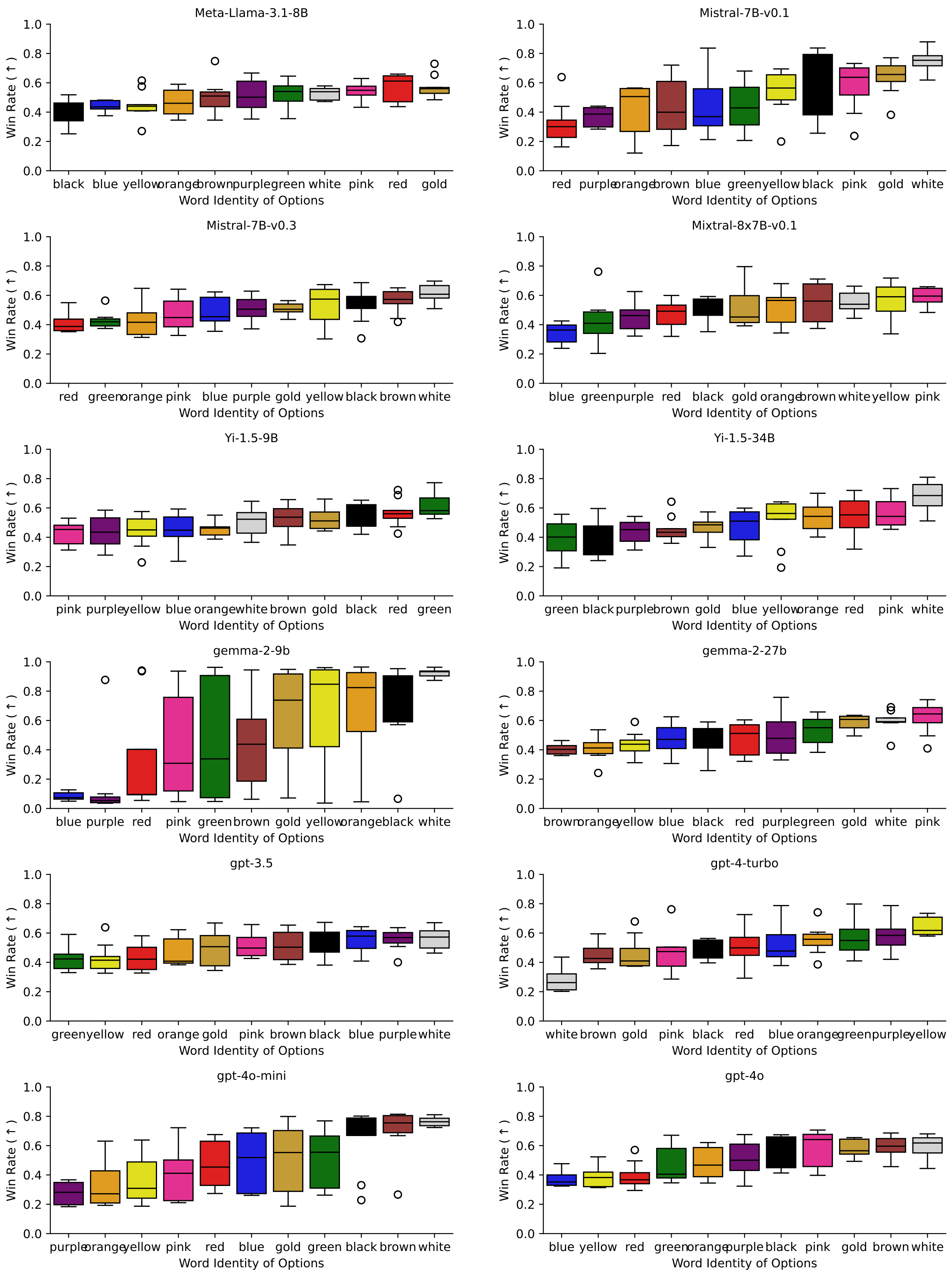

Figure 14:
Win rates correspond to option order.

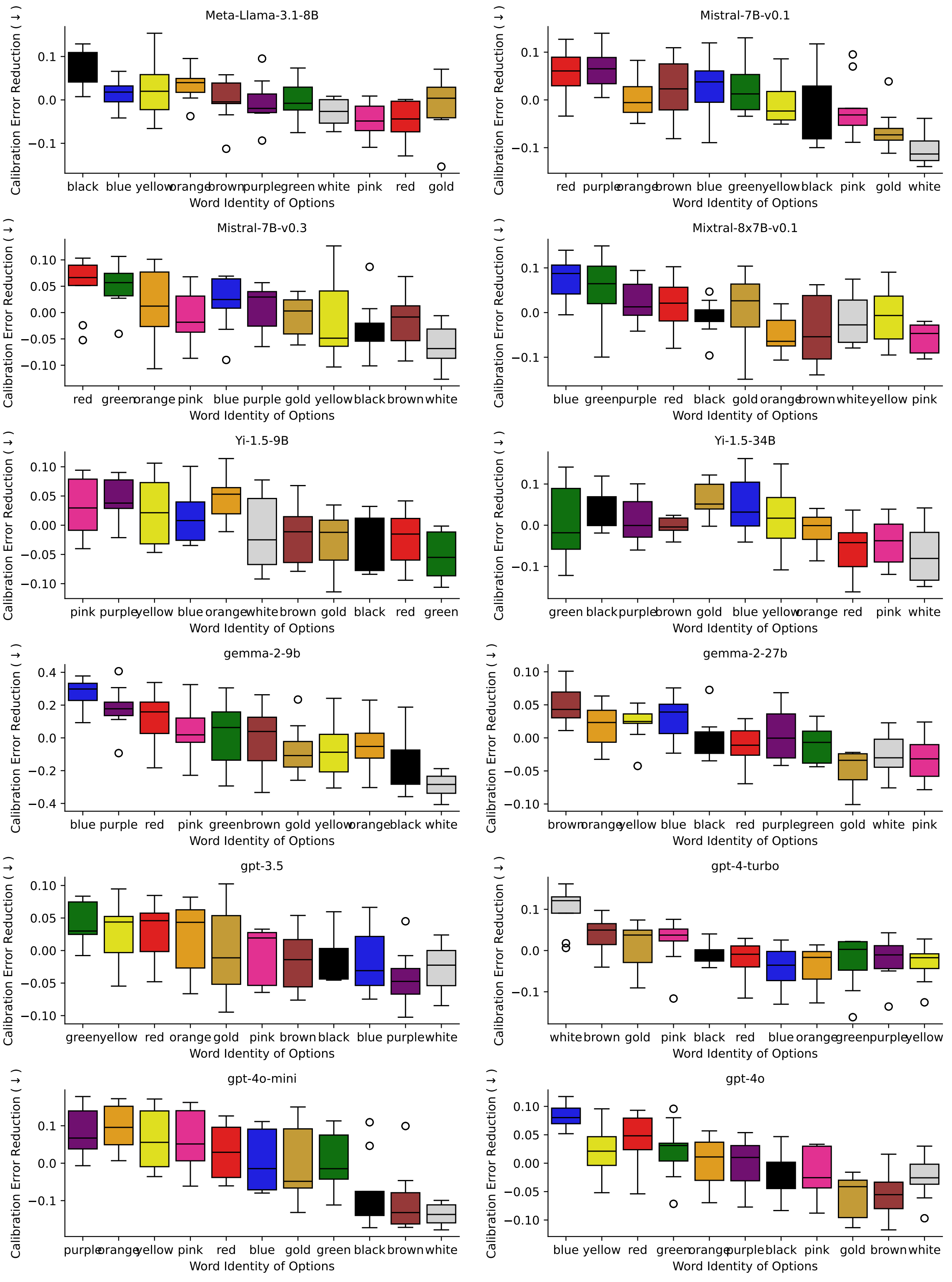

Figure 15: **Reductions in error that correspond to option order.**

		Spearman's Corr	PValue	Kendall's Tau	PValue
gpt-4-turbo	gpt-4o	-0.61	8.83e-13	-0.45	2.08e-06
gpt-4-turbo	gpt-4o-mini	-0.58	3.37e-11	-0.35	3.10e-04
gemma-2-9b	gpt-4-turbo	-0.49	6.90e-08	-0.27	4.41e-03
Mistral-7B-v0.1	Mixtral-8x7B-v0.1	-0.44	1.46e-06	-0.35	3.10e-04
gemma-2-27b	gpt-4-turbo	-0.40	1.51e-05	-0.42	1.27e-05
Mixtral-8x7B-v0.1	gpt-4-turbo	-0.36	1.10e-04	-0.16	8.76e-02
Meta-Llama-3.1-8B	gpt-4-turbo	-0.34	2.64e-04	-0.31	1.25e-03
Mistral-7B-v0.3	gemma-2-9b	-0.34	3.10e-04	-0.09	3.43e-01
Mistral-7B-v0.3	Yi-1.5-34B	-0.32	6.85e-04	-0.38	6.71e-05
Mixtral-8x7B-v0.1	Yi-1.5-9B	-0.30	1.41e-03	-0.20	3.68e-02
Mistral-7B-v0.3	gemma-2-27b	-0.29	1.89e-03	-0.38	6.71e-05
Meta-Llama-3.1-8B	gemma-2-9b	-0.26	6.73e-03	-0.35	3.10e-04
Mistral-7B-v0.1	Yi-1.5-34B	-0.25	7.97e-03	-0.31	1.25e-03
Yi-1.5-9B	gemma-2-9b	+0.26	6.30e-03	+0.20	3.68e-02
gpt-3.5	gpt-4o	+0.28	3.37e-03	+0.31	1.25e-03
Meta-Llama-3.1-8B	Yi-1.5-34B	+0.33	4.45e-04	+0.31	1.25e-03
Meta-Llama-3.1-8B	gemma-2-27b	+0.33	3.97e-04	+0.09	3.43e-01
gemma-2-9b	gemma-2-27b	+0.33	3.69e-04	+0.20	3.68e-02
gemma-2-27b	gpt-4o	+0.34	2.83e-04	+0.31	1.25e-03
Yi-1.5-9B	gpt-4o-mini	+0.42	4.90e-06	+0.42	1.27e-05
Mistral-7B-v0.1	Yi-1.5-9B	+0.42	4.08e-06	+0.27	4.41e-03
Mixtral-8x7B-v0.1	gemma-2-9b	+0.42	3.88e-06	+0.31	1.25e-03
gpt-4o-mini	gpt-4o	+0.46	4.08e-07	+0.24	1.36e-02
Mixtral-8x7B-v0.1	gpt-4o	+0.47	2.04e-07	+0.42	1.27e-05
Mistral-7B-v0.1	gpt-4o-mini	+0.48	1.50e-07	+0.56	3.99e-09
gemma-2-9b	gpt-4o-mini	+0.51	1.10e-08	+0.42	1.27e-05
Yi-1.5-34B	gemma-2-27b	+0.57	7.59e-11	+0.42	1.27e-05
gemma-2-9b	gpt-4o	+0.62	4.33e-13	+0.53	3.69e-08

Table 14: Correlation matrix between model rankings. We show only the statistically significant correlations, others had both lower effect sizes and were not significant. In most cases both Kendall Tau and Spearman were significant; there are two exceptions where for $p = 0.01$ Kendall Tau was not significant where Spearman was significant.

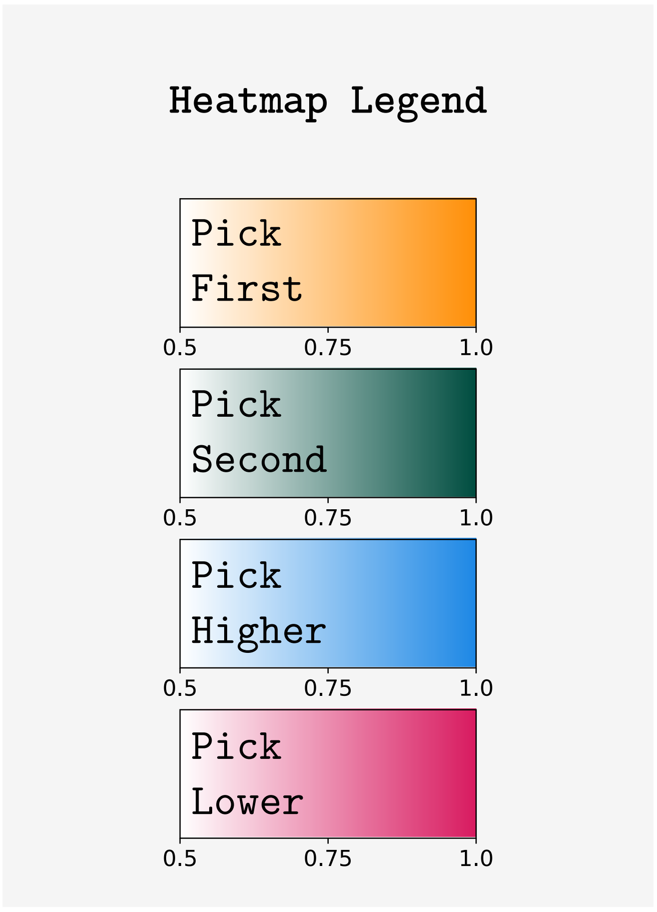

Figure 16: **Legend for heatmaps on the following page.**

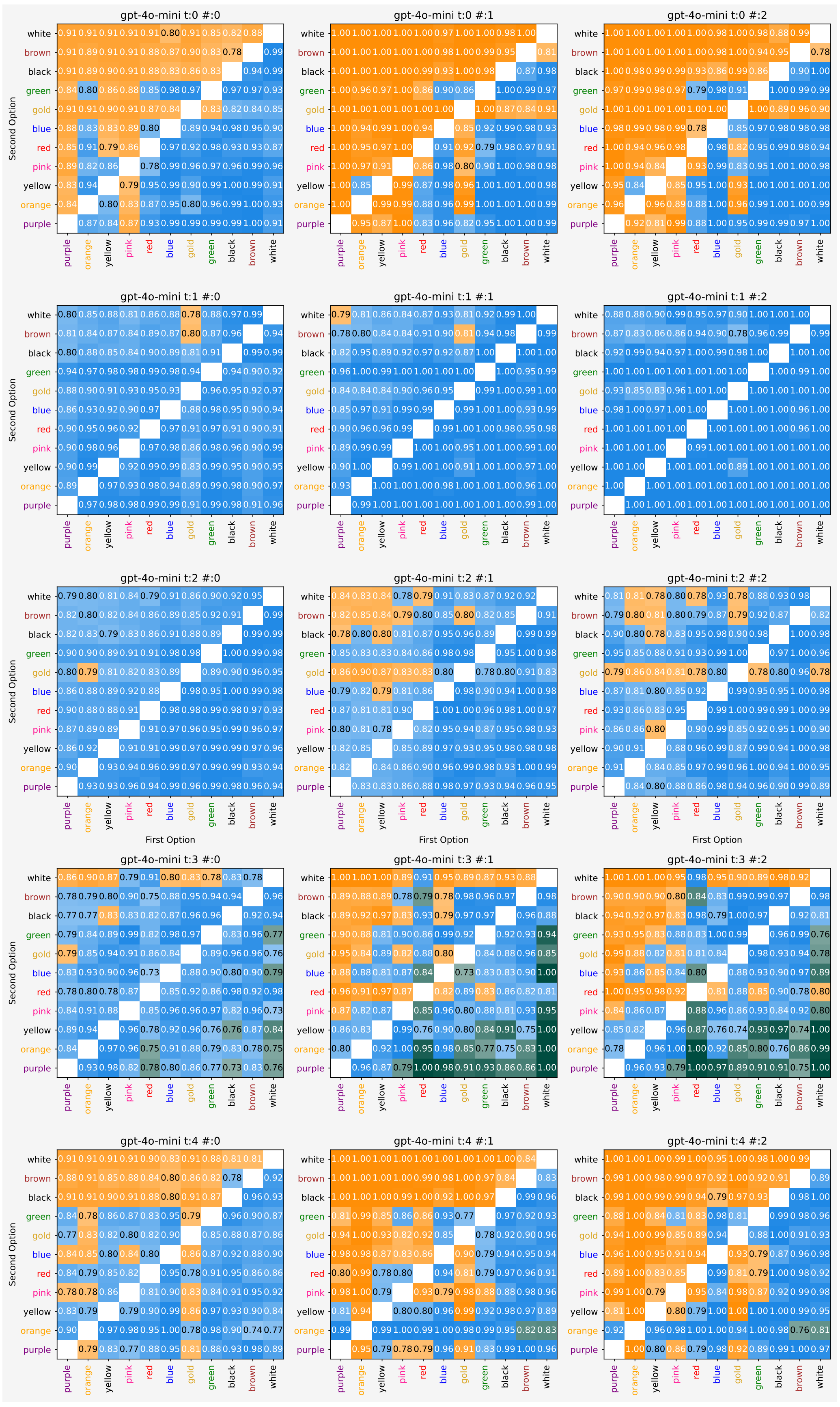
Figure 17: **Behavior Compatibility-based Heatmap for `gpt-4o-mini`.**

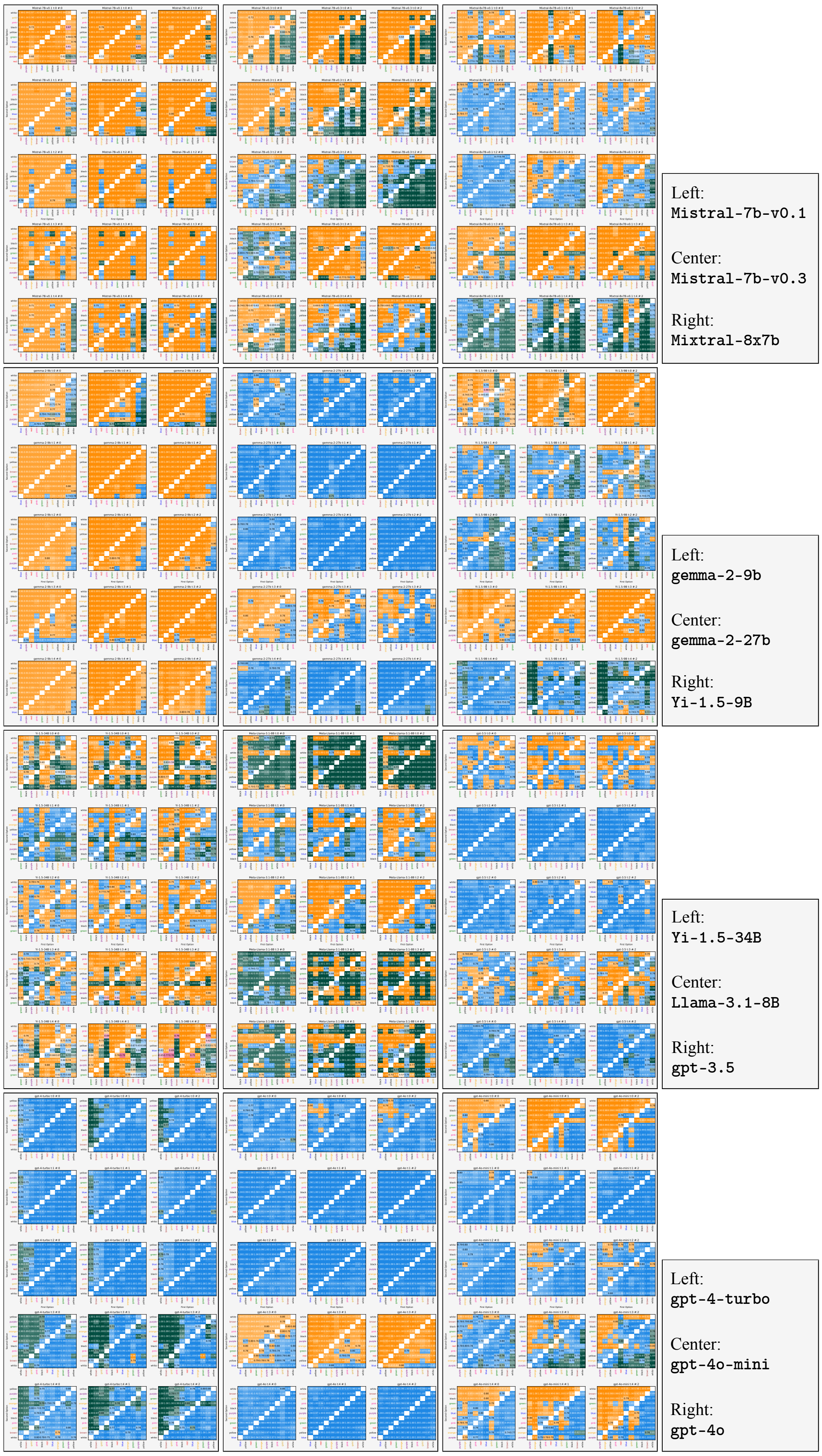

Figure 18: **Space Station View of Model Behaviors.**

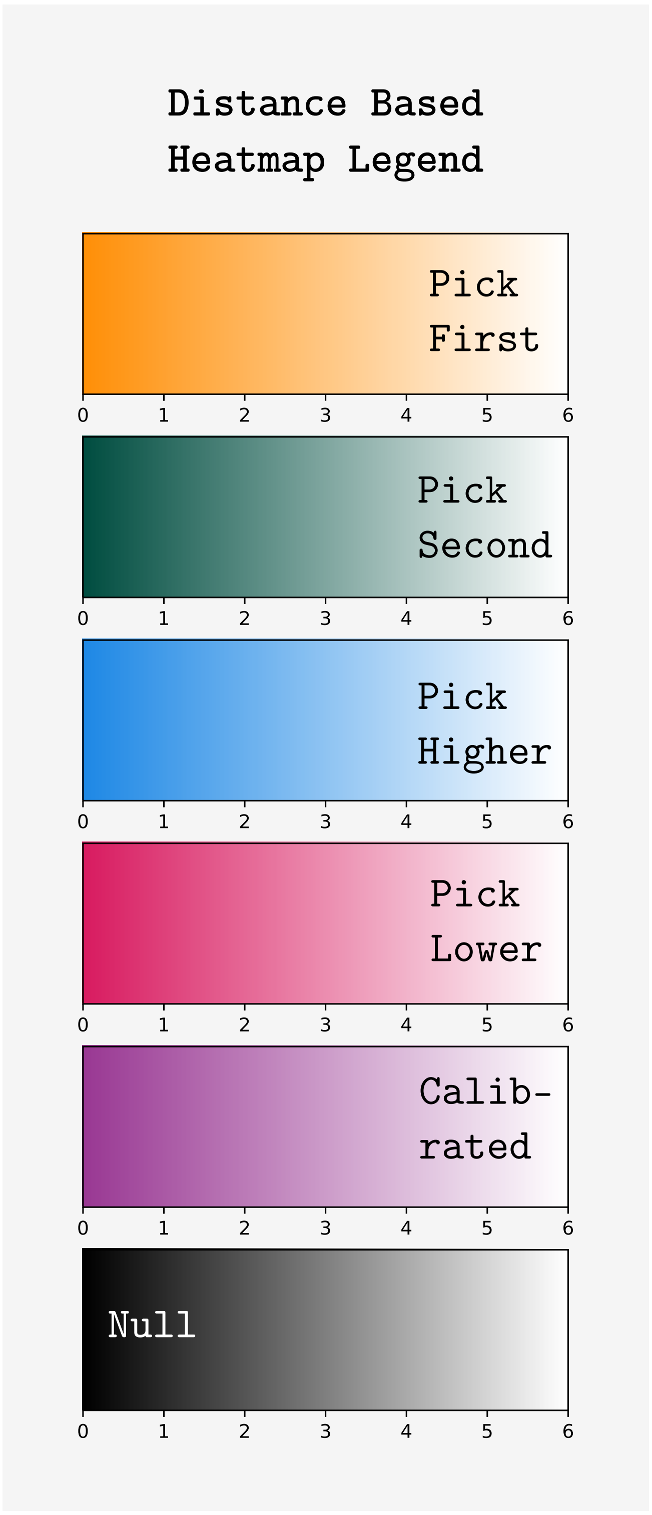

Figure 19: **Legend for heatmaps on the following page.**

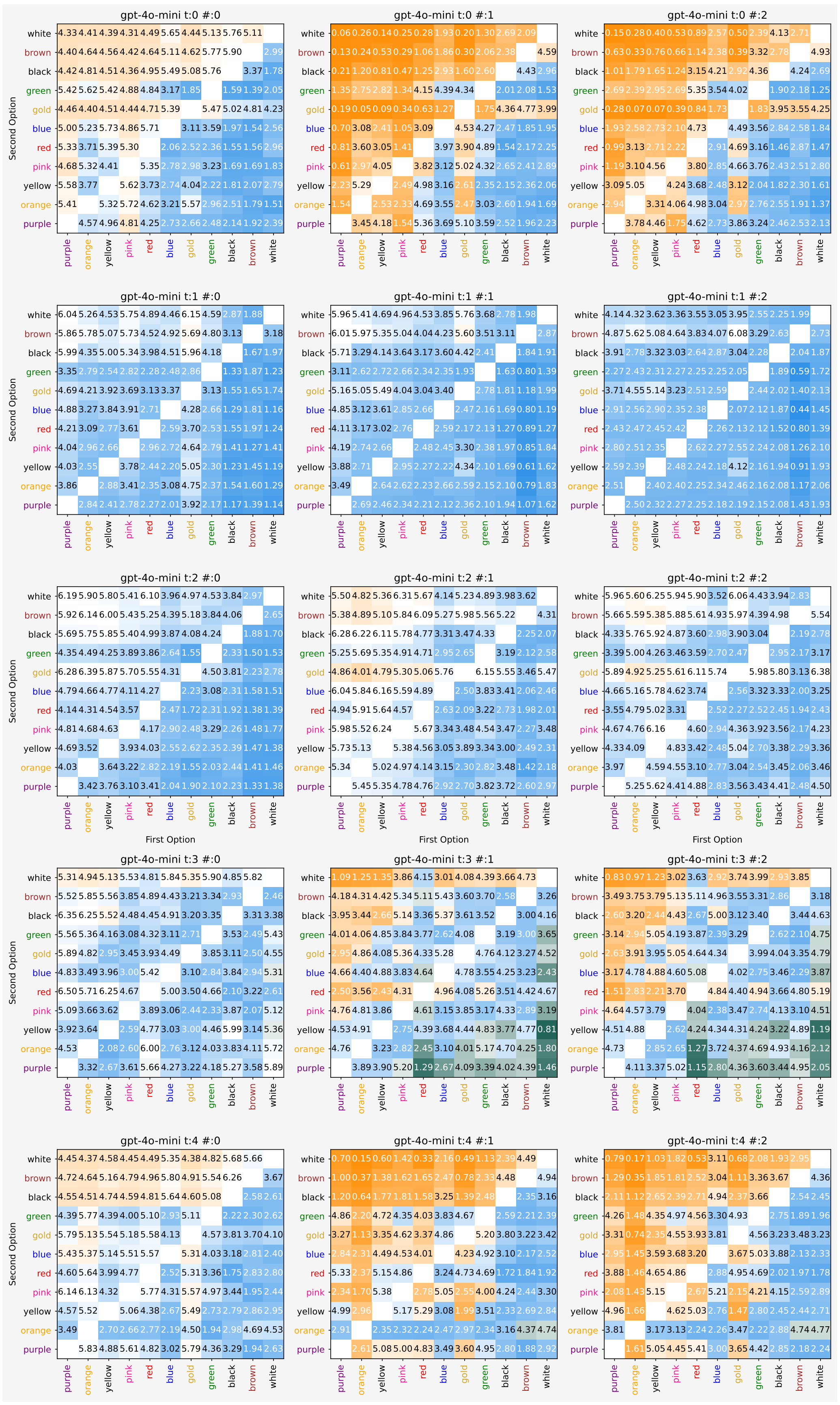

Figure 20: **Distance-based Heatmap for `gpt-4o-mini`.**

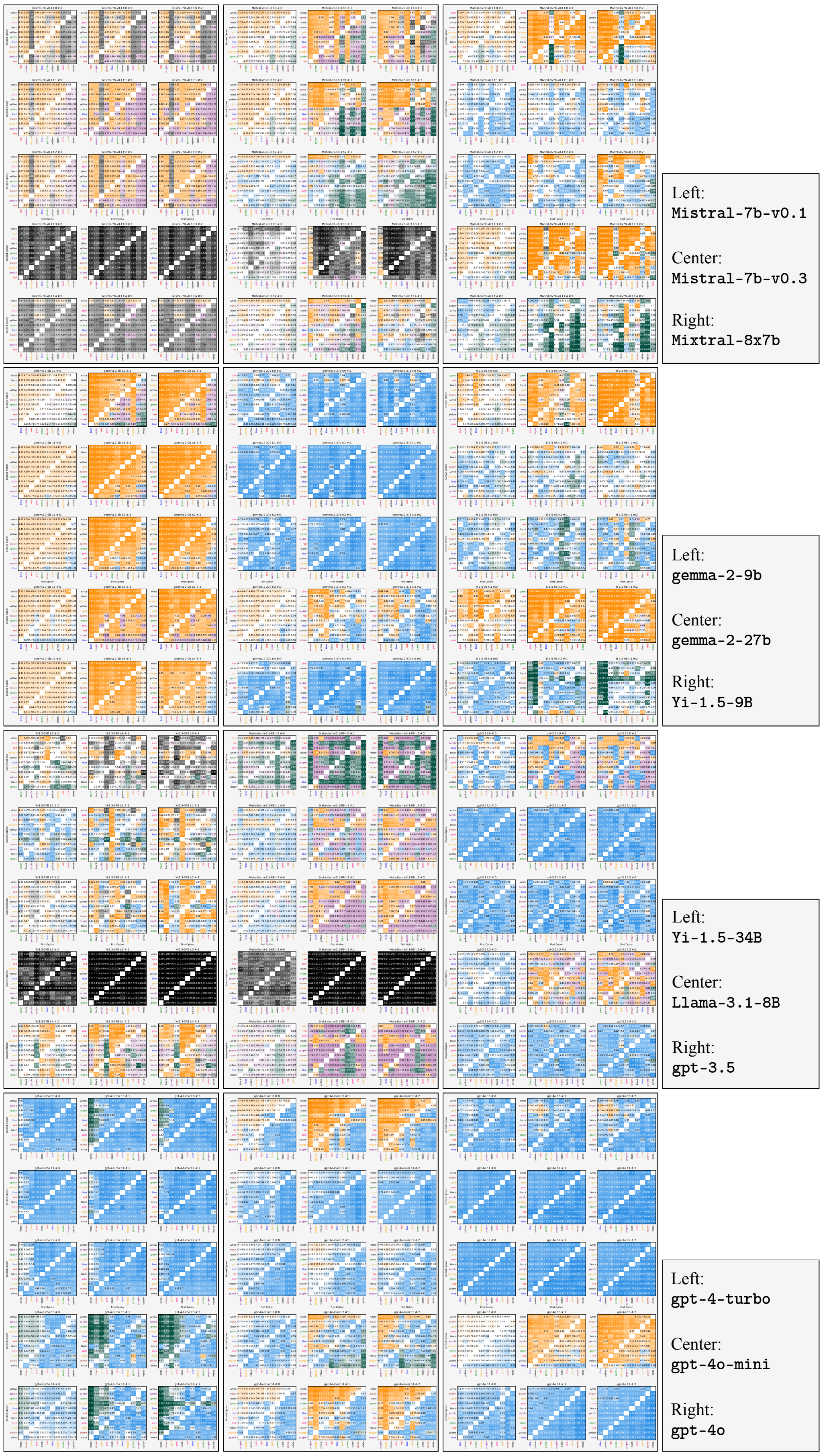

Figure 21: **Space Station View of Model Behaviors.**

Q Figures: More Results on Number Frequency

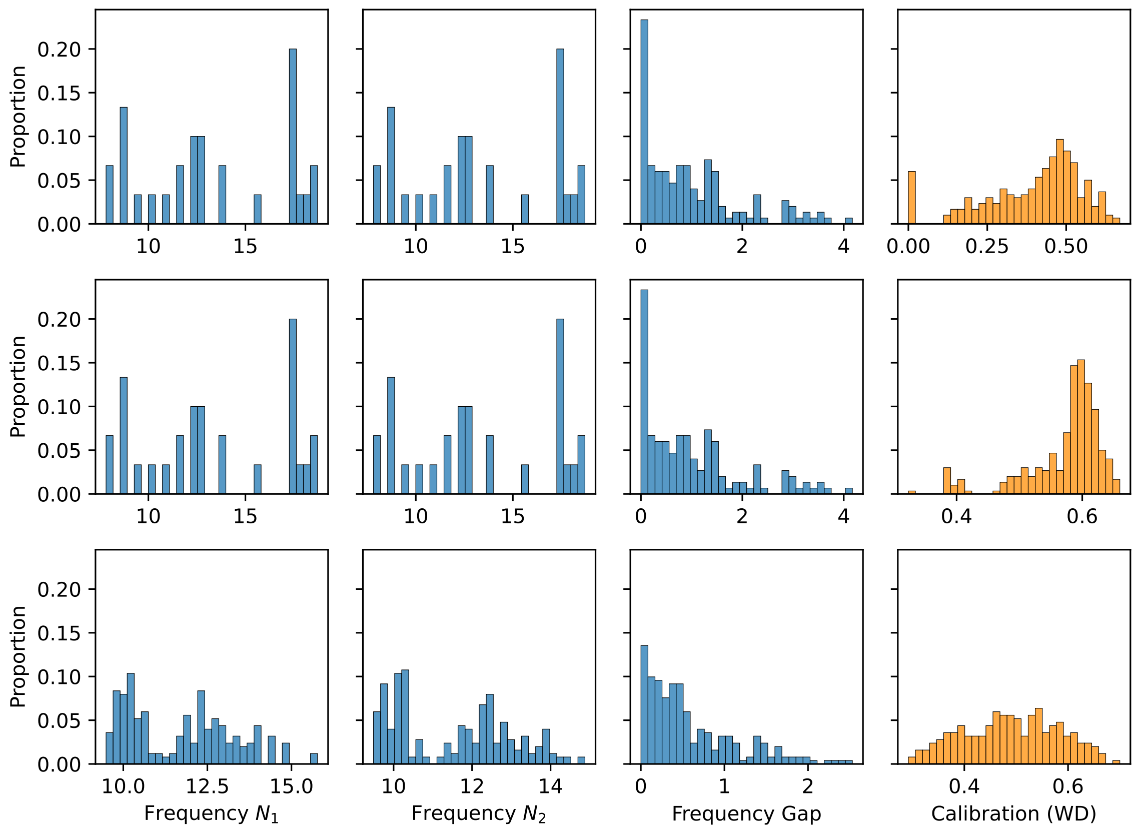

Figure 22: This plot shows dataset frequency statistics proportions for the number tokens used across the different datasets. We also present, highlighted orange, `gpt-4o`'s calibration score distribution.

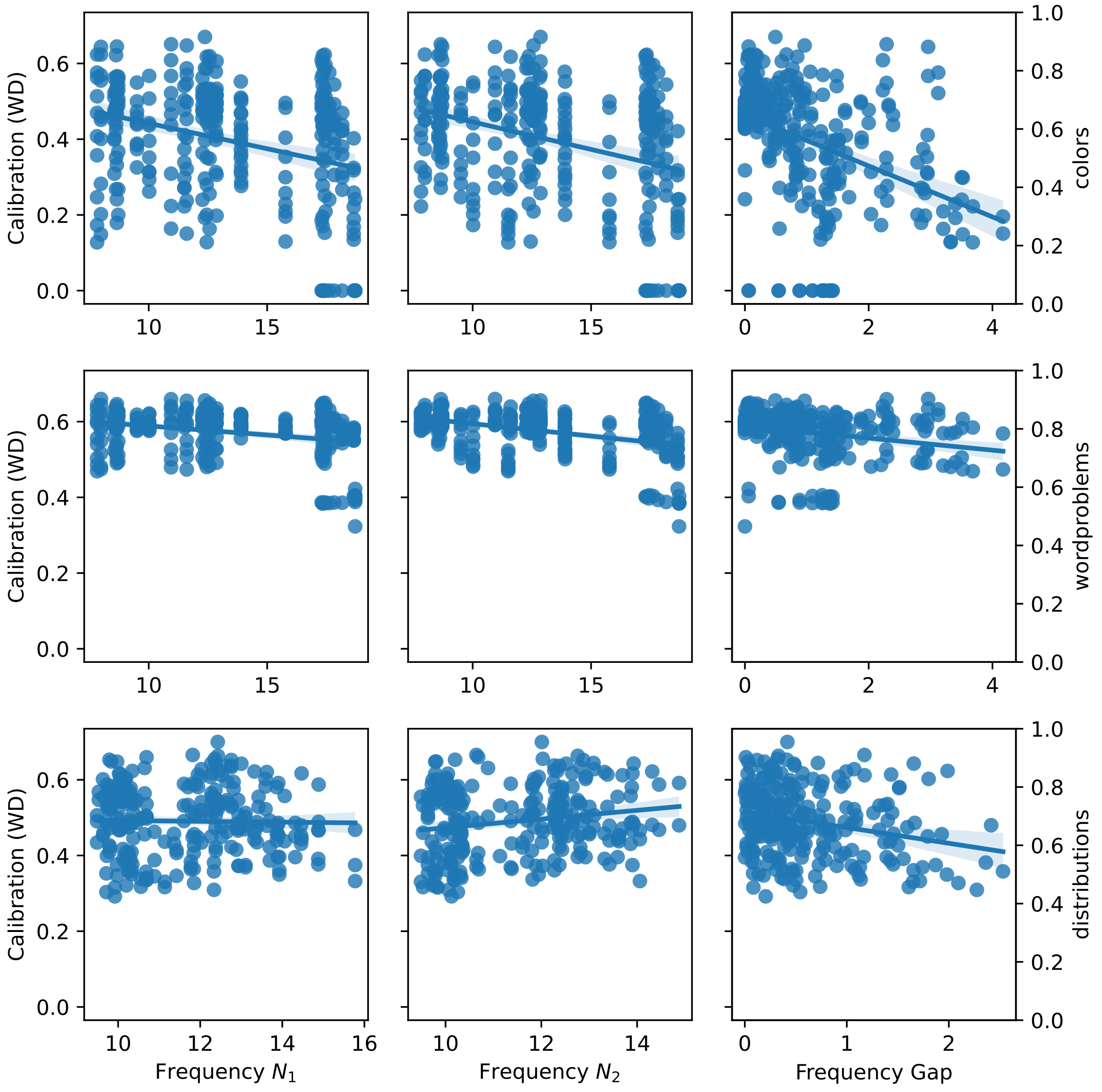

Figure 23: This plot shows `gpt-4o` data comparing calibration performance and different frequency data. Each row of results is across a different dataset.

R Figures: Human Experiments

```python
CoinFace = str, H: CoinFace = "H", T: CoinFace = "T"
def flip(face: CoinFace) -> CoinFace:
    return {H: T, T: H}[face]
def mdp_1gram(sequence: List[CoinFace]) -> CoinFace:
    return _mdp_ngram(sequence, 1)
def mdp_2gram(sequence: List[CoinFace]) -> CoinFace:
    return _mdp_ngram(sequence, 2)
def mdp_3gram(sequence: List[CoinFace]) -> CoinFace:
    return _mdp_ngram(sequence, 3)
def mdp_4gram(sequence: List[CoinFace]) -> CoinFace:
    return _mdp_ngram(sequence, 4)
def mdp_5gram(sequence: List[CoinFace]) -> CoinFace:
    return _mdp_ngram(sequence, 5)
def _mdp_ngram(sequence: List[CoinFace], ngram_size: int) -> CoinFace:
    """Picks the most common continuation for a given sequence based upon all n_grams of the
    ↪ current size.

    Defaults to H if sequence is empty. Flips previous coinface if the length of the sequence is
    ↪ 1.
    """
    assert ngram_size >= 1

    # Handle short sequences
    if len(sequence) == 0:
        return H
    elif len(sequence) == 1:
        if sequence[0] == H:
            return T
        else:
            return H
    if len(sequence) <= ngram_size:
        return _mdp_ngram(sequence, min(ngram_size - 1, len(sequence) - 1))

    map_ = {H: 0, T: 1}
    sequence_numbers = [map_[face] for face in sequence]
    mdp = np.zeros([2 for _ in range(ngram_size + 1)])
    for i in range(0, len(sequence_numbers) - ngram_size):
        index = tuple(sequence_numbers[i : i + (ngram_size + 1)])
        mdp[index] = mdp[index] + 1

    last = tuple(sequence_numbers[-ngram_size:])
    transitions = mdp[last]
    if transitions[0] > transitions[1]:
        return H
    elif transitions[0] == transitions[1]:
        # NOTE: Alt, we could pick which ever is more common
        # or fall back to a lower ngram or sample.
        return H
    else:
        return T
```

n-gram model implementation

Figure 24: **n-gram model implementation.**