Teaching models to express their uncertainty in words

Stephanie Lin *sylin07@gmail.com*
University of Oxford

Jacob Hilton *jhilton@openai.com*
OpenAI

Owain Evans *owaine@gmail.com*
University of Oxford

Reviewed on OpenReview: *https://openreview.net/forum?id=8s8K2UZGTZ*

Abstract

We show that a GPT-3 model can learn to express uncertainty about answers using natural language – without use of model logits. When given a question, the model generates both an answer and a level of confidence (e.g. "90% confidence" or "high confidence"). These levels map to probabilities that are *well calibrated*. The model also remains moderately calibrated under distribution shift, and is sensitive to uncertainty in its *own* answers, rather than imitating human examples.

For testing calibration, we introduce the CalibratedMath suite of tasks. We compare the calibration of uncertainty expressed in words ("verbalized probability") to uncertainty extracted from model logits. Both kinds of uncertainty are capable of generalizing calibration under distribution shift. We also provide evidence that GPT-3's ability to generalize calibration depends on pre-trained latent representations that correlate with epistemic uncertainty over its answers.

1 Introduction

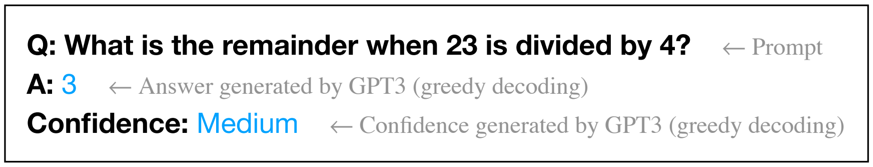
Current state-of-the-art language models perform well on a wide range of challenging question-answering tasks (Brown et al., 2020; Chowdhery et al., 2022; Hoffmann et al., 2022). They can even outperform the average human on the MMLU benchmark (which consists of exam-like questions across 57 categories) and on BIG-Bench (which consists of 150+ diverse tasks). Yet when models generate long-form text, they often produce false statements or "hallucinations" (Lin et al., 2021; Maynez et al., 2020; Shuster et al., 2021). This reduces their value to human users, as users cannot tell when a model is being truthful or not.

The problem of truthfulness motivates *calibration* for language models (Nguyen & O'Connor, 2015). If models convey calibrated uncertainty about their statements, then users know how much to trust a given statement. This is important for current models (which often hallucinate falsehoods) but also for any model that makes statements where there is no known ground truth (e.g. open problems in science or mathematics, or economic forecasts as in Zou et al. (2022)).

Previous work on calibration focuses on the model log-probabilities or "logits" (Guo et al., 2017; Jiang et al., 2021). Yet the log-probabilities of models like GPT-3 represent uncertainty over *tokens* (ways of expressing a claim) and not epistemic uncertainty over claims themselves. If a claim can be paraphrased in many different ways, then each paraphrase may have a low log-probability.[1] By contrast, when humans express uncertainty, this is epistemic uncertainty about the claim itself.[2] In this paper, we take a step toward teaching models to

[1]Sometimes it's feasible to sum over the probabilities of all paraphrases of a claim. But if the claim is complex, the space of possible paraphrases will be vast and hard to demarcate.

[2]If a human says "I think it's likely this vaccine will be effective", they express confidence about the *vaccine* not the string "vaccine".



Q: What is the remainder when 23 is divided by 4? ← Prompt

A: 3 ← Answer generated by GPT3 (greedy decoding)

Confidence: Medium ← Confidence generated by GPT3 (greedy decoding)

Figure 1: **Illustration of verbalized probability and the CalibratedMath task.** The prompt is in bold and GPT-3's output is in blue. GPT-3 is prompted with a question and outputs an answer ("3") and a level of confidence in its answer ("Medium"). GPT-3 is scored on the calibration of its confidence (not on the accuracy of its answer). In this example, the answer is correct but the confidence is only "Medium". Using our MSE metric (Section 2.3), this confidence would score $(1 - 0.5)^2 = 0.25$.

express epistemic uncertainty using natural language. We call this "verbalized uncertainty" or "verbalized probability".

The goal of verbalized probability is to express uncertainty as part of a model's natural language output without directly mimicking human training data. Models should be calibrated about the likelihood that their own statements are correct, which differs from the uncertainty that a human might have when making a similar statement. For example, GPT-3 outperforms most humans on a computer security quiz (Hendrycks et al., 2020) but is much worse than humans at arithmetic questions of the form "$2 \times 3 \times 7 =?$". Thus, we expect pre-trained models will need to be finetuned to produce calibrated verbalized probability.

We focus on the simple case of verbalized uncertainty where the model outputs a probability or "confidence" as a single token (Fig. 1). This case is a step towards more sophisticated expression of uncertainty in natural language. Moreover, this case relates to methods that finetune model logits for calibration (Jiang et al., 2021), which we also investigate in this paper.

Training models in verbalized probability is a component of making models "honest" (Evans et al., 2021; Askell et al., 2021a; Christiano, 2021). We define a model as *honest* if it can communicate everything it represents internally in natural language (and will not *misrepresent* any internal states). Honesty helps with AI alignment: if an honest model has a misinformed or malign internal state, then it could communicate this state to humans who can act accordingly. Calibration is compatible with a certain kind of dishonesty, because a model could be calibrated by simply imitating a calibrated individual (without having the same "beliefs" as the individual). However, if GPT-3 achieves good calibration on diverse questions after finetuning as in Section 3.1, it seems unlikely that it dishonestly misrepresents its confidence.

1.1 Contributions

We introduce a new test suite for calibration. CalibratedMath is a suite of elementary mathematics problems. For each question, a model must produce both a numerical answer and a confidence in its answer (see Figure 1). There are many types of question, which vary substantially in content and in difficulty for GPT-3. This allows us to test how calibration generalizes under distribution shifts (by shifting the question type) and makes for a challenging test (see Figure 3). Since GPT-3's math abilities differ greatly from humans, GPT-3 cannot simply imitate human expressions of uncertainty.

GPT-3 can learn to express calibrated uncertainty using words ("verbalized probability"). We finetune GPT-3 to produce verbalized probabilities. It achieves reasonable calibration both in- and out-of-distribution, outperforming a fairly strong baseline (Figure 5 and Table 1).

This calibration performance is not explained by learning to output logits. GPT-3 does not simply learn to output the uncertainty information contained in its logits (Section 3.4). We also show that certain superficial heuristics (e.g. the size of the integers in the arithmetic question) cannot explain the performance of verbalized probability.

Kind of probability	Definition	Example	Supervised objective	Desirable properties
Verbalized (number / word)	Express uncertainty in language ('61%' or 'medium confidence')	**Q: What is 952 – 55?** **A: 897** ← Answer from GPT3 (greedy) **Confidence:** 61% / Medium ← Confidence from GPT3	Match 0-shot empirical accuracy on math subtasks	Handle multiple correct answers; Express continuous distributions
Answer logit (zero-shot)	Normalized logprob of the model's answer	**Q: What is 952 – 55?** **A:** 897 ← Normalized logprob for GPT3's answer	None	Requires no training
Indirect logit	Logprob of 'True' token when appended to model's answer	**Q: What is 952 – 55?** **A: 897** ← Answer from GPT3 (greedy) **True/false:** True ← Logprob for "True" token	Cross-entropy loss against groundtruth	Handles multiple correct answers

Figure 2: **Three kinds of probability used in this paper.** Prior work on calibration focuses on the answer logit. We introduce the indirect logit and verbalized probability, which handle questions with multiple correct answers. Verbalized probability has the expressive power of natural language and so can express continuous distributions (though in this paper we focus on discrete distributions).

We compare verbalized probability to finetuning the model logits. We show how to finetune GPT-3 to express epistemic uncertainty via its model logits (see "Indirect logit" in Table 2) and find that this also generalizes calibration under distribution shift (Table 1).

2 Setup

2.1 Calibration and Three Kinds of Probability

We want to test the calibration of language models for uncertainty over their own answers to questions. The basic idea is that if a calibrated model assigns 90% to an answer, then the answer is correct 90% of the time. Formally, let M be a model, q be a question, a_M be the model's answer, and $p_M = \Pr(a_M|q)$ be the assigned probability that a_M is correct. Then these assigned probabilities are (perfectly) calibrated if:

$$\Pr(a_M|p_M = p) = p \tag{1}$$

for $p \in [0, 1]$ (Guo et al., 2017). In this paper, we test calibration on different sets of questions to evaluate how well calibration generalizes under distribution shift (Ovadia et al., 2019).

We consider three sources for the probability p_M that the model's answer is correct, as shown in Figure 2. Two of the kinds of probability ("answer logit" and "indirect logit") are based on the log-probabilities that a language model assigns to tokens. Thus they cannot be used for models without a tractable likelihood on outputs (e.g. information retrieval models that call out to external resources). By contrast, verbalized probabilities apply to any model that outputs natural language. Moreover, verbalized probabilities mirror human expression of uncertainty. This allows models to respond to prompts from non-technical users (e.g. "How sure are you about what you just said?", "I've told you my confidence on a scale from 1-5. Can you do the same?"). This also allows models to decide when and how to provide uncertainty information (depending on the human audience).

2.2 CalibratedMath

CalibratedMath is a test suite consisting of 21 arithmetic tasks, including addition, multiplication, rounding, arithmetic progressions, and finding remainders (see full details in Table 3). For each task, questions and answers are programmatically generated. The answers are always integers and for some tasks there are multiple correct answers (e.g. "Name any prime number below 208?"). The 21 tasks are further divided

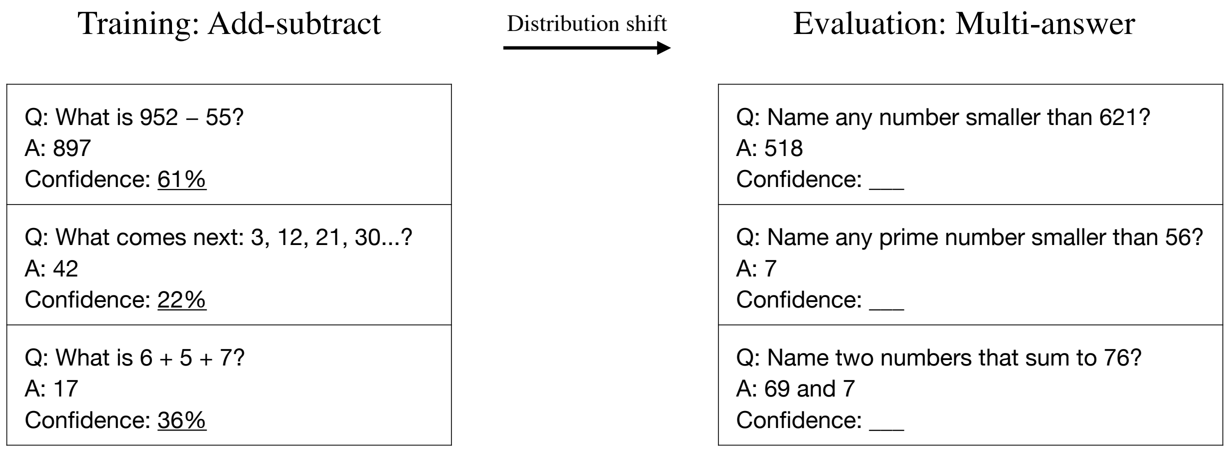


Figure 3: **Examples from training and one of the evaluation sets for CalibratedMath.** GPT-3 is finetuned on the Add-subtract training set (left). Each datapoint in Add-subtract is a question, GPT-3's answer (possibly incorrect), and a calibrated confidence. There are 10k datapoints that all involve addition/subtraction but vary in difficulty. Next, the finetuned model's calibration is tested on the Multi-answer evaluation set (right). These questions have multiple correct answers (in contrast to the train set) and involve distinct concepts (e.g. prime numbers). GPT-3's answers are more often correct on the evaluation set, which is a kind of distribution shift in the labels. (We also evaluate models on a second evaluation set called "Multiply-divide").

into sub-tasks based on the number of digits in each operand and the number format. The sub-tasks vary in difficulty for GPT-3. For example, multiplication is harder than addition and gets more difficult as the number of digits is increased. The fact that some sub-tasks are predictably easier or harder for GPT-3 is crucial for a challenging test of calibration.

As in prior work on calibration in ML (Ovadia et al., 2019; Karandikar et al., 2021), we focus on how well calibration *generalizes* under distribution shift. Our main experiments use the "Add-subtract" training set (Figure 3). This consists of tasks in CalibratedMath that involve addition or subtraction and have a unique correct answer. The evaluation set (called "Multi-answer") consists of questions with multiple correct answers that sometimes involve multiplication and division. There is a distribution shift between training and evaluation, with the following two aspects:

- **Shift in task difficulty:** GPT-3 is more likely to answer questions in the evaluation set (Multi-answer) correctly than the training set (Add-subtract). Median accuracy is 65% for Multi-answer and 21% for Add-subtract (for full details see Figure 8). Thus, to be well calibrated, the model should assign higher probabilities on average to answers in the evaluation set than the training set. This is essentially a shift in the "label distribution" from training to evaluation. (We expect language models other than GPT-3 to have a similar distribution shift for the same reason.)
- **Shift in content:** The training and evaluation sets differ in the mathematical concepts they employ and whether or not there are multiple correct answers.

Though not shown in Figure 3, models trained on Add-subtract are also evaluated on a second evaluation set called "Multiply-divide". Questions in Multiply-divide have unique correct answers but are more difficult than those in Add-subtract and include distinct concepts related to multiplication and division (Table 3).

2.3 Metrics

Our goal is to measure the model's calibration when expressing uncertainty about its own zero-shot answers. In all our experiments, the model's zero-shot answers are held fixed. The goal is not to improve the model's answers but instead to improve calibration in expressing uncertainty over these answers.[3] Calibration is measured using two metrics:

Mean squared error (MSE). Following Section 2.1, for each question the model M assigns a probability p_M to its own answer a_M being correct. The MSE compares p_M to the groundtruth of whether a_M is correct or not:

$$\mathbb{E}_q[(p_M - \mathbb{I}(a_M))^2]$$

Note that a model can be perfectly calibrated (per Equation 1) and not have a MSE of zero. The MSE combines calibration error with "sharpness" (Kuleshov & Liang, 2015), while the MAD (below) just measures the former. (The MSE is called the "Brier Score" in probabilistic forecasting.)

Mean absolute deviation calibration error (MAD). The MAD estimates how closely the model approximates Equation 1 based on a finite sample. Model probabilities are divided into K bins with equal numbers of samples, so the bins have denser coverage where there are more samples (Nguyen & O'Connor, 2015). Within each bin b_i, we calculate the proportion of correct answers ("acc(b_i)" or "accuracy") and average probability assigned to answers in b_i ("conf(b_i)" or the "average confidence"). Then the MAD is given by:

$$\frac{1}{K}\sum_{i=1}^{K} |\mathrm{acc}(b_i) - \mathrm{conf}(b_i)|$$

While this is not a proper scoring rule, it offers a simple numeric summary of the calibration curves shown in Section 3 (Hendrycks et al., 2018; Nixon et al., 2019).

3 Experiments

For our experiments, we used the 175-billion parameter GPT-3 model ("davinci") via the OpenAI API (Brown et al., 2020). We tried out smaller models but their performance on arithmetic questions is too weak for CalibratedMath to be challenging.[4]

How can we finetune a pre-trained model to output calibrated verbalized probabilities? We finetune GPT-3 using supervised learning. This approach is less principled and flexible than using reinforcement learning (with rewards derived from a proper scoring rule). However, supervised learning was easier to implement using OpenAI's API, and provides an interesting test of generalization outside the training distribution.

3.1 Supervised finetuning

To finetune GPT-3 to produce verbalized probabilities, we need a labeled training set. Each input is a question followed by GPT-3's answer and the label is a (calibrated) confidence (see Figure 3). The basic intuition is that for questions GPT-3 is likely to get wrong, its confidence should be low. Thus, we use GPT-3's *empirical accuracy* on each type of question as the label. We recognize that this approach can lead to suboptimal labels. For example, it might use a low-confidence label for "$10 \times 10 = 100$" because most two-digit multiplications are hard for GPT-3. But we will show that the approach works well enough for our purposes.

Formally, let q be a question from sub-task T. Let a_M be GPT-3's answer to q. We define $\hat{p}_T$ associated with the input (q, a_M) to be GPT-3's empirical accuracy on sub-task T:

$$\hat{p}_T = \mathbb{E}_{q \in T}[\mathbb{I}(a_M)]$$

[3]In general, training a model to improve calibration may also improve the accuracy of the model's answers. However, for CalibratedMath, the training we provide for calibration is unlikely to improve accuracy very much. Thus, it's reasonable to measure calibration with respect to the zero-shot answers even after finetuning.

[4]We tested smaller models including GPT-J (Wang & Komatsuzaki, 2021) and the 7B-parameter GPT-3 on the arithmetic questions. Their performance is so weak that guessing 0% for every question would achieve reasonable calibration. To learn more about how different models perform on CalibratedMath, we recommend using models comparable to GPT-3-175B in performance.

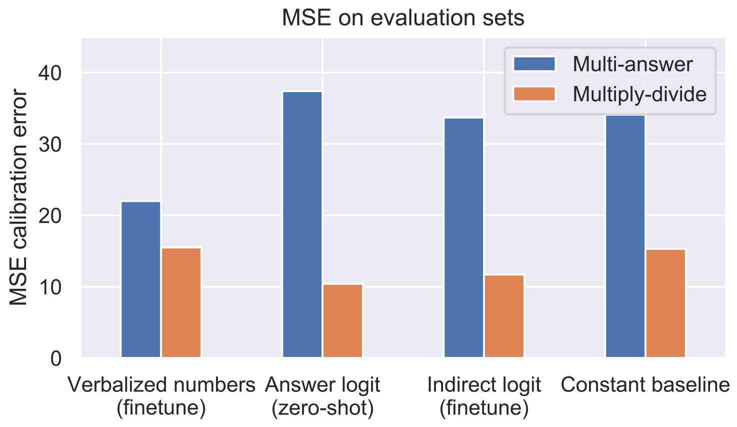


Figure 4: **Calibration scores on the Multi-answer and Multiply-divide evaluation sets.** The same results are shown in Table 1 below.

which we estimate using random samples generated from T. The full training set is then constructed as follows. For each sub-task T we randomly sample 100 questions and generate GPT-3's zero-shot answers (using greedy decoding) for a total of $|T| \times 100 \approx 10$k inputs. We then compute the $\hat{p}_T$ for each T and use it to construct the label for each sample from T.

The label is a simple transformation of $\hat{p}_T$. For the "verbalized numbers" setup, the label is given by $\lfloor 100 * \hat{p}_T \rfloor$. In the "verbalized words" setup, we use a set of five words (e.g. "lowest", "low", "medium", "high", "highest") to express the degree of confidence. We map $\hat{p}_T$ to one of five words corresponding to probability intervals of width 0.2. Categories can then be mapped back to probability values by taking the midpoint of the corresponding interval. (We found that using meaningful words – such as "lowest" etc. – worked slightly less well than meaningless names. See Appendix B.1.)

3.1.1 Indirect logit and baselines

For the indirect logit (defined in Figure 2), we use the same random sample of 100 questions from each sub-task (along with GPT-3's zero-shot answer). However, in this case the label for each individual question-answer pair is the boolean True/False value indicating whether the model's answer was correct, for which we have the groundtruth. Thus we can optimize the cross-entropy loss. Further details for the supervised finetuning setup are given in Appendix B.3.

We compare the two finetuned setups (verbalized probability and indirect logit) to the "zero-shot answer logit" (see Fig. 2). We also include a "constant baseline". This baseline uses a constant probability on the evaluation set, where the value of the constant is the best-scoring value on the training set (in terms of MSE)[5]. Metrics are shown in Table 1 and Figure 4, while calibration curves are in Figure 5.

3.2 Results

Verbalized probability generalizes well to both eval sets. The main result is shown in Table 1 and Figures 4 and 5. After finetuning on the Add-subtract training set, verbalized probabilities generalize reasonably well to both the Multiply-divide and Multi-answer evaluation sets. So the model remains moderately calibrated under a substantial distribution shift. In terms of MSE, the model outperforms the two logit setups on Multi-answer and matches the constant baseline on Multiply-divide.[6] We ran an additional experiment to probe generalization, where we flipped around the training set (training on Multiply-divide and evaluating on both Add-subtract and Multi-answer). Again, verbalized probability generalizes reasonably

[5]For the constant baseline, the MAD is the difference in model accuracy between training and evaluation tasks.

[6]The shift in task difficulty from Add-subtract to Multiply-divide is relatively small. So the constant baseline should do reasonably well in MSE (and very well in MAD).

Table 1: **Calibration scores on evaluation sets.** The finetuned setups were trained on the Add-subtract set. We test how well calibration generalizes under distribution shift. Scores are in percentage terms and lower is better. Note: the MSE is not for answers to questions but for the probability the answers are correct.

Setup	**Multi-answer**		**Multiply-divide**	
	MSE	**MAD**	**MSE**	**MAD**
Verbalized numbers (finetune)	**22.0**	**16.4**	15.5	19.0
Answer logit (zero-shot)	37.4	33.7	**10.4**	9.4
Indirect logit (finetune)	33.7	38.4	11.7	**7.1**
Constant baseline	34.1	31.1	15.3	8.5

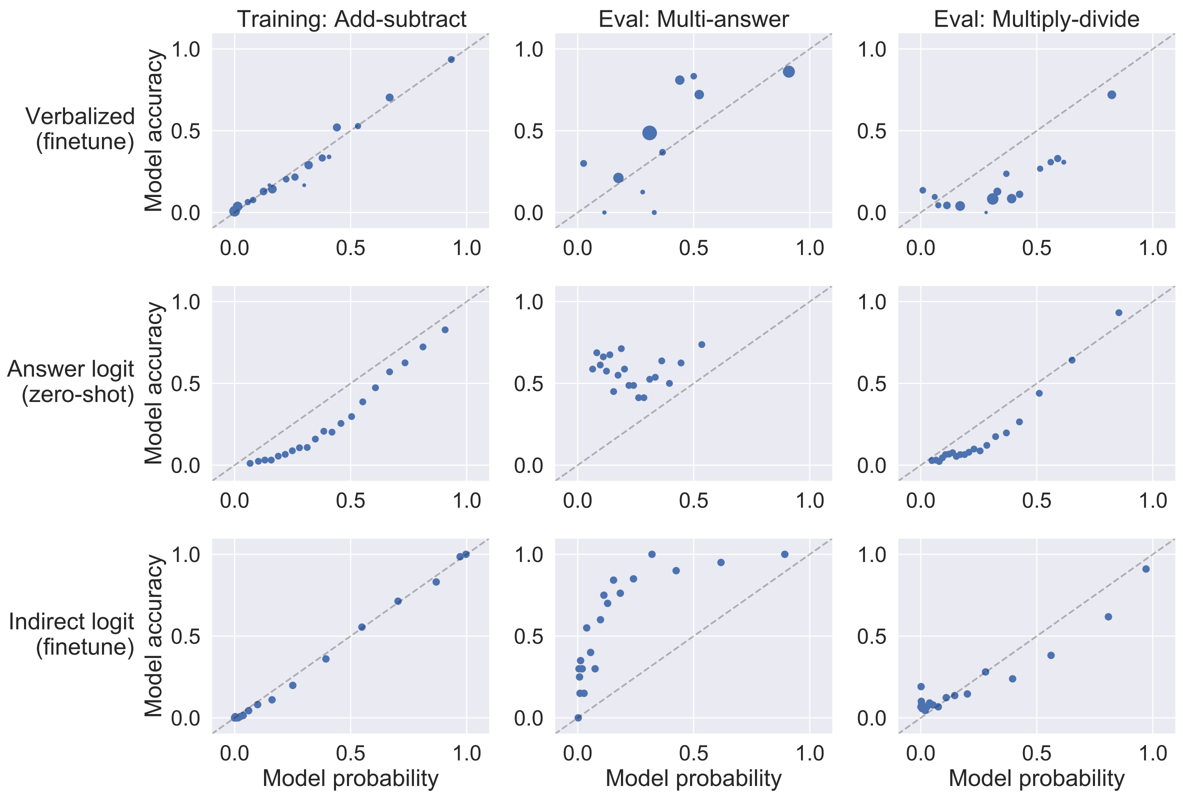


Figure 5: **Calibration curves for training (left) and evaluation (center and right).** Curves are generated using the same procedure as the MAD (Section 2.3). The probabilities for each question are divided into bins, and the y-value for a bin is the proportion of questions for which the answer was true (i.e. the model accuracy). The size of markers indicates the bin size. We see that the two logit setups are very *underconfident* on the Multi-answer evaluation, while all three setups are better calibrated on the Multiply-divide evaluation.

well and outperforms other setups on Multi-answer (see Appendix C.3). Finally, we find that verbalized probability performs similarly whether the model outputs tokens for words or numbers (see Appendix C.4).

Verbalized probability overfits to training. Calibration for verbalized probability is much better in-distribution. The model is underconfident in its answers to Multi-answer because these answers are more likely to be correct than those for the Add-subtract training set.[7]

[7]Our results suggest that the finetund GPT-3 will only output a verbal probability (e.g. 96%) if that precise token ("96%") appeared during training. This would explain the lack of smoothness in the calibration curves in Figure 5.

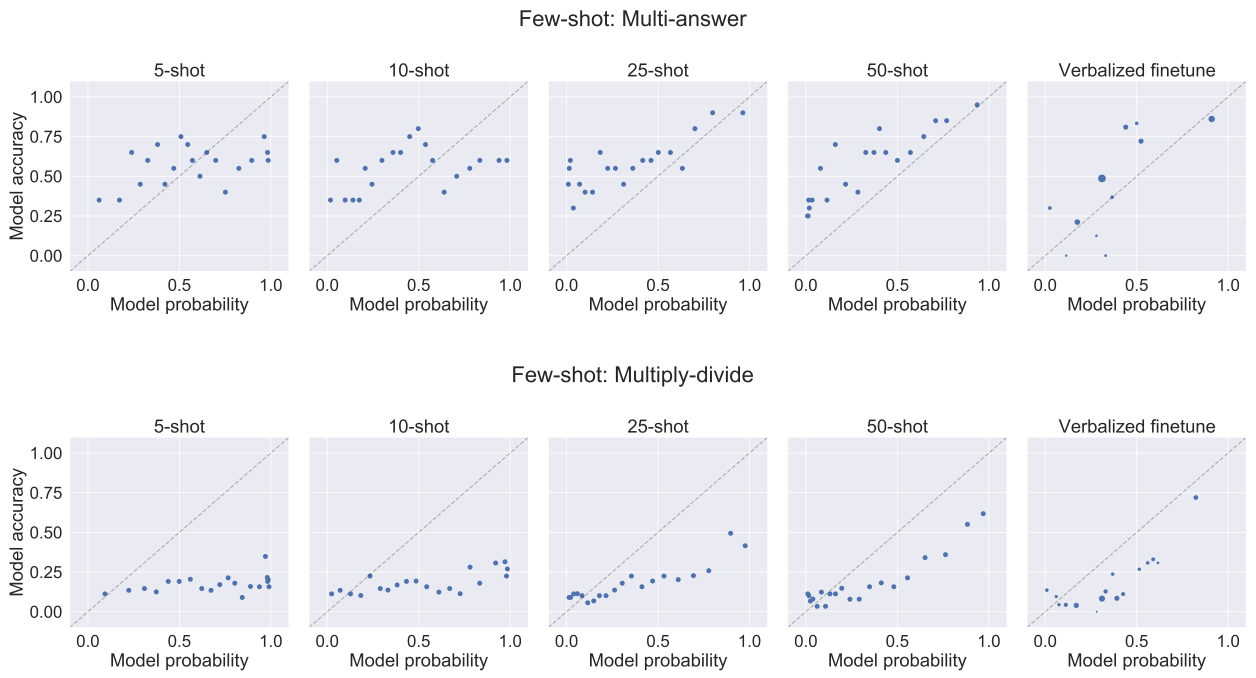


Figure 6: **Calibration curves for few-shot learning (verbalized probability).** Compares stochastic k-shot for varying k (using Expected Value decoding) to supervised finetuning (10k datapoints with greedy decoding) on the evaluation sets. 50-shot is almost as calibrated as the finetuned setup.

Indirect logit generalizes well to Multiply-divide. The indirect logit achieves impressive calibration on the Multiply-divide evaluation set, where it outperforms other models. However, it does worse than verbalized probability on the Multi-answer evaluation. This is likely because it is more difficult to avoid overfitting given our setup.[8] Further work could explore how the indirect logit compares to verbalized probability with different training setups (e.g. a more diverse distribution on probabilities and questions).

3.3 Stochastic Few-shot

In order to learn more about how verbalized probability generalizes, we tested GPT-3's calibration in a stochastic k-shot setting, while varying k from 1 to 50. We used the following procedure. For each question in the evaluation set, we randomly sample k new examples from the Add-subtract training set and include them in the context.[9] In order to generate verbalized probabilities, we do not use greedy decoding (as in the finetuning experiments) but instead find the weighted sum of the model's top five tokens (where the weights are the model probabilities for the tokens). This "Expected Value decoding" is less in the spirit of verbalized probabilities, but gives us a sense of the model's capabilities (see Appendix C.2). The resulting calibration curves are shown in Figure 6.

On both evaluation sets, GPT-3 starts out visibly uncalibrated, but begins to show improvement at $k = 25$ and above. At $k = 50$, performance is already close to that of the finetuned models, which are trained on over 2.5k samples. One potential explanation is that GPT-3 already has latent representations for questions and answers that relate to calibrated confidence, and the few-shot examples allow it to locate the task (Reynolds & McDonell, 2021). We discuss this in the following section.

[8]It's possible to do early stopping for verbalized probability by stopping when the actual MSE on the training set stops decreasing – but this is not available for the indirect logit (Appendix B.3).

[9]If we used a fixed set of k examples, the model tends to mimic the most recent example in the prompt – leading to high variance.

3.4 Explaining the performance of verbalized probability

We have shown that GPT-3 learns to express uncertainty in words and generalize calibration to new tasks. But what exactly has GPT-3 learned and would the learned features enable generalization beyond our experiments?

Does GPT-3 just learn to output the logits? One possibility is that the verbalized probability results are fully explained by GPT-3 learning to output information in its logits. However, we have already seen that verbalized probability generalizes better than the answer logit on the Multi-answer evaluation. Moreover, on the Multiply-divide evaluation, the correlation in performance between verbalized probability and answer logit across sub-tasks is only modest (see Appendix C.4). So GPT-3 must be using more than just the information in the logits.

Does GPT-3 just learn simple heuristics (e.g. low probability for questions with large integers)? Another possibility is that verbalized probability results are explained by GPT-3 learning simple heuristics for the difficulty of questions. For example, suppose GPT-3 simply learned to output lower probabilities for questions with larger integers (because they are more difficult). This would not lead to robust generalization, as some questions with small integers are difficult. We ran an experiment to test whether simple heuristics can generate calibrated probabilities. We trained a logistic regression model on the Add-subtract training set with the same target probabilities as in Section 3.1. The model has hand-crafted features that we know are predictive of difficulty for GPT-3: the number of digits of integers in the question, the operator (e.g. "+" or "`round to nearest 10`"), and the number format (e.g. "1000" or "1,000"). This heuristic model performed worse than verbalized probability on both the Multi-answer and Multiply-divide evaluation sets (Table 2). So the results for verbalized probability cannot be fully explained by these heuristics.

Evidence that GPT-3 uses latent (pre-existing) features of questions. So what does explain GPT-3's ability to generalize calibration? There is tentative evidence that GPT-3 learns to use features of inputs that it already possessed *before* finetuning. We refer to these features as "latent" representations, because they are not "active" in pre-trained GPT-3 (which is poorly calibrated).

Via OpenAI's Embeddings API (Neelakanta, 2022), we can extract an embedding for each question-answer pair in CalibratedMath using a GPT-3 model finetuned for semantic similarity.[10] Figure 7 shows a (trained) projection of GPT-3's embeddings into two dimensions on the Multiply-divide evaluation set, where we see that samples are already reasonably well separated into correct and incorrect classes. Since a linear 2D projection is able to uncover this structure, we view this as evidence that the embedding already encoded features that were relevant to calibration.

The "Linear probe" row in Table 2 explores this further by attaching a linear probe to GPT-3's embeddings and predicting whether GPT-3's embedded answer was correct or incorrect. While performance is worse than the finetuned verbalized model, the probe still exhibits generalization to the Multiply-divide evaluation set, again indicating that GPT-3 learned relevant features during pre-training that are now present in the embedding.

Finally, from Section 3.3, GPT-3 is able to generalize its calibration on both evaluation sets after seeing only $k = 50$ examples. Given the high number of tasks and difficulty levels in CalibratedMath, a context containing 50 examples can only cover a tiny fraction of the space of inputs. It would therefore be difficult to meta-learn new features that would generalize robustly to the evaluation sets.

4 Discussion

4.1 Directions for future work

Our results show that GPT-3 has some ability to generalize (verbalized) calibration under distribution shift. However, while our training and evaluation sets differed significantly in the label distribution, the content and format of questions did not shift much. Future work could test whether calibration generalizes to

[10]While the embeddings come from a finetuned GPT-3 model, we expect the results would be similar if embeddings came from the pre-trained model.

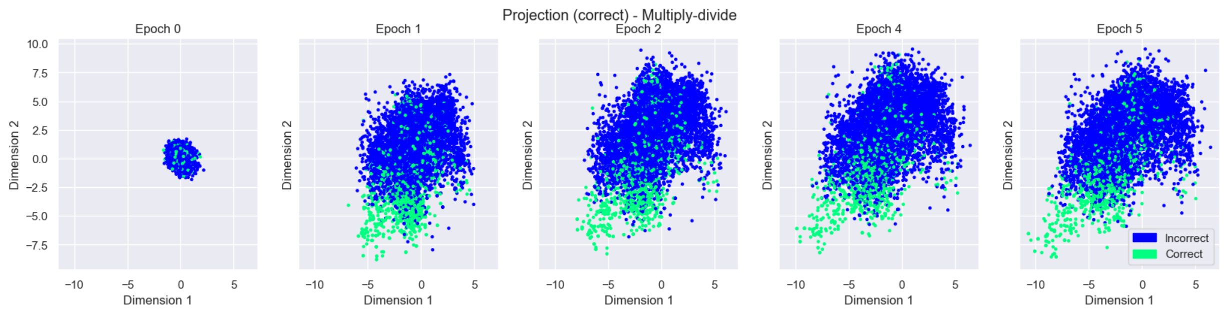


Figure 7: **Linear projection of GPT-3 embeddings into two dimensions with colors denoting true (green) or false (blue).** Each point is the embedding of an input pair of form `(question, GPT-3 answer)` from the Multiply-divide evaluation set that has been projected into 2D. A point is green if the GPT-3 answer is correct and blue otherwise. We see the classes become better separated as training progresses and after 5 epochs they are reasonably well separated by a linear boundary.

Table 2: **Calibration performance of alternative models.** Verbalized probability outperforms simple heuristics, but the linear probe on pre-trained embedding model performs well.

Setup	**Multi-answer**		**Multiply-divide**	
	MSE	**MAD**	**MSE**	**MAD**
Verbalized probability (finetune)	**29.0**	**24.0**	**12.7**	**10.6**
Log. reg. with heuristic features	29.7	31.2	17.7	18.5
Linear probe on GPT3 embedding	31.2	30.1	14.0	14.2

other subject areas (e.g. history or biology) and to other formats (e.g. chat, long-form question answering, forecasting).

Using models with the same capacity as GPT-3 but different architectures and pre-training would allow experiments into cross-model calibration. If a model with significantly different internal representations can give similarly calibrated uncertainties for GPT-3's statements, this would raise questions about whether GPT-3 is actually expressing its own epistemic state. While we attempted this experiment using the 7B-parameter GPT-3 model, the difference in capacity (as observed in performance on arithmetic tasks) was a significant confounder. (For related experiments see Kadavath et al. (2022), which came out after this paper.)

5 Related work

Calibration in new domains. Prior work on calibration focuses primarily on the classification setting, where models output a probability distribution over the set of possible classes (Guo et al., 2017; Mukhoti et al., 2020; Minderer et al., 2021), corresponding to what we call the "answer logit". To generalize calibration to a new target domain, methods often require samples from the target or from additional source domains (Gong et al., 2021; Csurka, 2017; Wang et al., 2021). We study how calibration generalizes when a pre-trained model is finetuned on a single source domain and must generalize zero-shot to a new domain.

Pre-trained language models. Hendrycks et al. (2020) analyze GPT-3's behavior on a benchmark of tasks that vary in both subject matter and difficulty, showing that GPT-3's calibration (for the answer logit) generalizes fairly poorly in both the zero-shot and few-shot settings. To improve the calibration of pre-trained language models, Desai & Durrett (2020) use label smoothing to reduce overconfidence on out-of-domain data. Kong et al. (2020) introduce on- and off-manifold regularization to handle in-distribution and out-of-distribution calibration, respectively, but focus on OOD detection rather than generalization. Other work focuses on the closely related problem of teaching models to abstain from answering when a model has high uncertainty about its answer. Kamath et al. (2020) train an auxiliary "calibrator" to predict whether the primary model correctly answers any given question using a mix of in-domain and out-of-domain data. In cases where the calibrator predicts an error, the model can refuse to answer. Additional studies explore the use of manually crafted prompts that instruct models to defer or qualify their answers when uncertain (Askell et al., 2021b; Lin et al., 2021). These methods typically correct for models being *overconfident* on out-of-domain examples. In comparison, GPT-3's accuracy on our target domain is much higher than its accuracy on the source domain; its predictions therefore tend to be *underconfident*. The shift between target and source is also much larger, where we move from a single-answer to a multi-answer setting.

Natural language generation. In the specific case of natural language generation, Jiang et al. (2021) study calibration by framing multiple-choice and extractive QA as generative tasks, where a language model's uncertainty can be extracted from its logits over all tokens in an answer sequence. The authors introduce methods for both fine-tuning and post-hoc calibration of logits. To handle answers that can be worded in more than one way, a round-trip translation model is used to generate paraphrases for each answer, and the model's uncertainty is calculated as its total probability across all such paraphrases. While this approach leads to better calibration, it adds additional overhead and doesn't handle the situation where a question has multiple answers that can't be exhaustively listed.

Verbalized uncertainty. Branwen (2020) demonstrates GPT-3's ability to express verbalized uncertainty on simple trivia questions in the in-domain, few-shot setting, using an instructive prompt. Mielke et al. (2022) also study verbalized uncertainty (referred to as "linguistic confidence" or "verbalized metacognition"). Their setup differs from ours in that they train an *external* calibrator to predict whether a given language model's answers are correct as a binary classification problem, and then use the calibration to guide the language model's generations such that they express the appropriate level of confidence. In contrast, we use the same model to both return answers and to give verbalized confidences on a numerical scale.

Acknowledgments

We thank William Saunders, Dan Hendrycks, Mark Xue, Jeff Wu, Paul Christiano, Daniel Ziegler, Collin Burns and Rai (Michael Pokorny) for helpful comments and discussions. We thank the anonymous reviewers at TMLR for detailed and constructive feedback.

References

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language assistant as a laboratory for alignment, 2021a.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language assistant as a laboratory for alignment, 2021b. URL `https://arxiv.org/abs/2112.00861`.

Gwern Branwen. Gpt-3 nonfiction - calibration, 2020. `https://www.gwern.net/GPT-3-nonfiction#calibration`, Last accessed on 2022-04-24.

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. Language models are few-shot learners, 2020.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Paul Christiano. ARC's first technical report: Eliciting latent knowledge, 2021. `https://www.alignmentforum.org/posts/qHCDysDnvhteW7kRd/arc-s-first-technical-report-eliciting-latent-knowledge`, Last accessed on 2022-04-30.

Gabriela Csurka. Domain adaptation for visual applications: A comprehensive survey, 2017. URL `https://arxiv.org/abs/1702.05374`.

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

Owain Evans, Owen Cotton-Barratt, Lukas Finnveden, Adam Bales, Avital Balwit, Peter Wills, Luca Righetti, and William Saunders. Truthful AI: Developing and governing AI that does not lie. *arXiv preprint arXiv:2110.06674*, 2021. URL `https://arxiv.org/abs/2110.06674`.

Yunye Gong, Xiao Lin, Yi Yao, Thomas G. Dietterich, Ajay Divakaran, and Melinda Gervasio. Confidence calibration for domain generalization under covariate shift. 2021. doi: 10.48550/ARXIV.2104.00742. URL `https://arxiv.org/abs/2104.00742`.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure, 2018. URL `https://arxiv.org/abs/1812.04606`.

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

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering. *Transactions of the Association for Computational Linguistics*, 9:962–977, 09 2021. ISSN 2307-387X. doi: 10.1162/tacl_a_00407. URL `https://doi.org/10.1162/tacl_a_00407`.

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

Amita Kamath, Robin Jia, and Percy Liang. Selective question answering under domain shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5684–5696, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.503. URL `https://aclanthology.org/2020.acl-main.503`.

Archit Karandikar, Nicholas Cain, Dustin Tran, Balaji Lakshminarayanan, Jonathon Shlens, Michael C Mozer, and Becca Roelofs. Soft calibration objectives for neural networks. *arXiv preprint arXiv:2108.00106*, 2021.

Lingkai Kong, Haoming Jiang, Yuchen Zhuang, Jie Lyu, Tuo Zhao, and Chao Zhang. Calibrated language model fine-tuning for in- and out-of-distribution data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1326–1340, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.102. URL `https://aclanthology.org/2020.emnlp-main.102`.

Volodymyr Kuleshov and Percy S Liang. Calibrated structured prediction. *Advances in Neural Information Processing Systems*, 28, 2015.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*, 2021. URL `https://arxiv.org/abs/2109.07958`.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661*, 2020.

Sabrina J. Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872, 2022. doi: 10.1162/tacl_a_00494. URL `https://aclanthology.org/2022.tacl-1.50`.

Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 15682–15694. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper/2021/file/8420d359404024567b5aefda1231af24-Paper.pdf`.

Jishnu Mukhoti, Viveka Kulharia, Amartya Sanyal, Stuart Golodetz, Philip Torr, and Puneet Dokania. Calibrating deep neural networks using focal loss. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15288–15299. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/aeb7b30ef1d024a76f21a1d40e30c302-Paper.pdf`.

Arvind Neelakanta. Introducing text and code embeddings in the openai api, 2022. `https://openai.com/blog/introducing-text-and-code-embeddings/`, Last accessed on 2022-04-30.

Khanh Nguyen and Brendan O'Connor. Posterior calibration and exploratory analysis for natural language processing models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1587–1598, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1182. URL `https://aclanthology.org/D15-1182`.

Jeremy Nixon, Mike Dusenberry, Ghassen Jerfel, Timothy Nguyen, Jeremiah Liu, Linchuan Zhang, and Dustin Tran. Measuring calibration in deep learning, 2019. URL `https://arxiv.org/abs/1904.01685`.

OpenAI. Fine-tuning, 2021. `https://beta.openai.com/docs/guides/fine-tuning/advanced-usage`, Last accessed on 2022-04-30.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift.* Curran Associates Inc., Red Hook, NY, USA, 2019.

Laria Reynolds and Kyle McDonell. Prompt programming for large language models: Beyond the few-shot paradigm, 2021. URL `https://arxiv.org/abs/2102.07350`.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*, 2021.

Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pp. 4627–4635. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/628. URL `https://doi.org/10.24963/ijcai.2021/628`. Survey Track.

Andy Zou, Tristan Xiao, Ryan Jia, Joe Kwon, Mantas Mazeika, Richard Li, Dawn Song, Jacob Steinhardt, Owain Evans, and Dan Hendrycks. Forecasting future world events with neural networks. *arXiv preprint arXiv:2206.15474*, 2022.

A CalibratedMath

Table 3: **Breakdown of tasks in the CalibratedMath benchmark.** '# Levels' refers to the count of difficulty levels within each operation, where the difficulty is determined by the number of digits in each operand and the formatting used for the numbers. Models are trained on tasks from the 'Add/Sub' group, then evaluated on either the 'Mult/Div' or the 'Multi[-answer]' group.

Group	**Operation**	**# Levels**	**Example**
Add/Sub	Addition	24	Q: What is 14 + 27? A: 41
Add/Sub	Subtraction	24	Q: What is 109 - 3? A: 106
Mult/Div	Multiplication	9	Q: What is 8 * 64? A: 512
Mult/Div	Division	12	Q: What is 512 / 8? A: 64
Mult/Div	Floor division	12	Q: What is 515 / 8? A: 64
Mult/Div	Modulo	12	Q: What is 515 mod 8? A: 3
Mult/Div	Remainder	12	Q: What is the remainder when 515 is divided by 8? A: 3
Mult/Div	Percentages	6	Q: What is 25% of 1024? A: 256
Mult/Div	Fraction reduction	7	Q: What is 15/24 in reduced form? A: 5/8
Add/Sub	Rounding	6	Q: What is 10,248 rounded to the nearest 10? A: 10,250
Add/Sub	Arithmetic sequences	6	Q: What comes next: 4, 14, 24, 34...? A: 44
Add/Sub	3-step addition	1	Q: What is 2 + 3 + 7? A: 12
Mult/Div	3-step multiplication	1	Q: What is 2 * 3 * 7? A: 42
Add/Sub	Addition (alt)	24	Q: What is 10 more than 23,298? A: 23,308
Add/Sub	Subtraction (alt)	24	Q: What is 24 less than 96? A: 72
Multi	Less than	2	Q: Name any number smaller than 100? A: 37
Multi	Greater than	2	Q: Name any number larger than 100? A: 241
Multi	Prime	2	Q: Name any prime number smaller than 100? A: 7
Multi	Square	2	Q: Name any perfect square smaller than 100? A: 64
Multi	Two-sum	2	Q: Name two numbers that sum to 25? A: 11 and 14
Multi	Multiple	6	Q: Name a single multiple of 7 between 80 and 99? A: 91

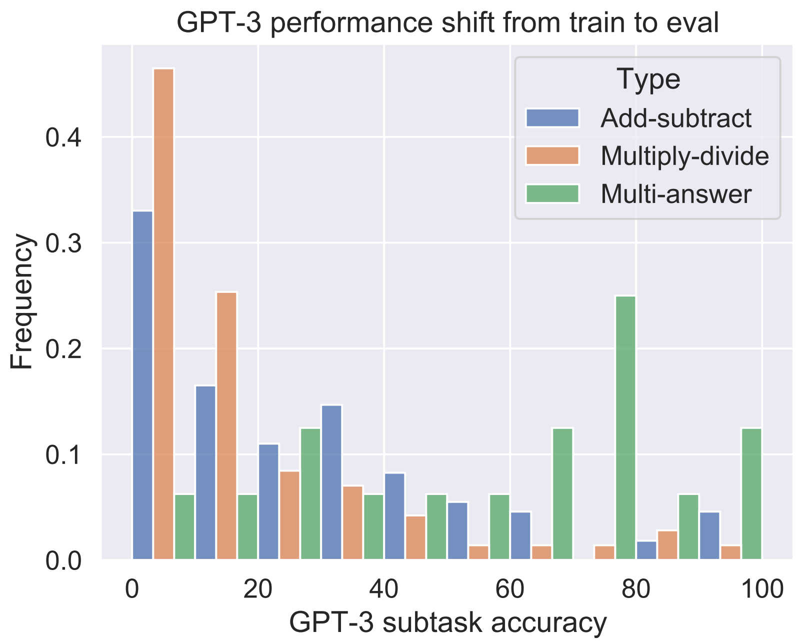


Figure 8: **Distribution shift of GPT-3's zero-shot ability to answer arithmetic questions between training (Add-subtract) and evaluation sets (Multi-answer and Multiply-divide).** For the training set "Add-subtract", we calculate the accuracy (% of correct answers) across each task and level of difficulty (see Table 3) and display this as a histogram. We see that the most frequent accuracies are close to 0 (which are question types such that GPT-3 gets nearly all instances wrong). The same process is repeated for the evaluation sets (Multi-answer and Multiply-divide). We see that GPT-3 does even worse on Multiply-divide but does much better on Multi-answer. Thus to be well calibrated on the Multi-answer evaluation set, GPT-3 would need to use higher probabilities (on average) than on the training set.

B Experimental setup

B.1 Verbalized probability with words

In one version of verbalized probability, models express uncertainty using words rather than numbers (see Figure 1 for an example). This leaves the question of which words to use for supervised finetuning. While we tried ordered categories (Confidence: “lowest”, “low”, “medium”, “high”, “highest”), we found that using random names without explicit orderings (“john”, “sam”, “matt”, “dan”, “tom”) led to very slightly better performance. So we use these random names throughout.

B.2 Prompts

```
Q: What is 57368 rounded to the nearest 100?
A: 57,400
Confidence: 19%

Q: What is 7 less than 58?
A: 51
Confidence: 44%

Q: What is 877 + 47?
A: 924
Confidence: 59%

Q: What is 517 - 898?
A: -381
Confidence: 67%

Q: What is 247 less than 4895?
A: 2352
Confidence: 0%

Q: What is 5 * 145?
A: 725
Confidence:
```

Figure 9: **Few-shot prompt.** The example shows a 5-shot prompt. The answers and target probabilities come from the estimation step described in Section 3. The prompt is randomized before every query.

B.3 Supervised fine-tuning

The supervised fine-tuning dataset consists of approximately 10k examples, where 100 examples are sampled from each sub-task in the training set. Models are trained for one epoch to prevent overfitting, using the default hyperparameters from OpenAI’s fine-tuning API with `learning_rate_multiplier = 0.1` (OpenAI, 2021). We additionally carry out a form of early stopping that takes into account the difference between the sub-task level targets $\hat{p}_T$, and a model’s binary accuracy of 0/1 on any individual question.

Consider a sub-task T from which we sample two questions, the first of which the model answers correctly. Then $\hat{p}_T$ would equal 0.5. If the model correctly gives uncertainties of 1 and 0 on the two samples, its per-sample MSE would be 0. However, it would incur a loss against the target $\hat{p}_T$. Reducing this loss would lead to *worse* performance on the per-sample MSE. This happens because $\hat{p}_T$ is a proxy for what the model’s uncertainty should be on any given question. As we continue to fit to $\hat{p}_T$, we see that per-sample MSE flattens or increases on the training set, even though the loss against $\hat{p}_T$ continues to decrease. We use this as a signal to stop training after around $n = 2700$ examples. A comparison of calibration by the number of samples seen is shown in Figure 11 on the two evaluation sets, although we use the training set only to determine the stopping point.

C Additional results

C.1 Verbalized calibration curves by number of training samples

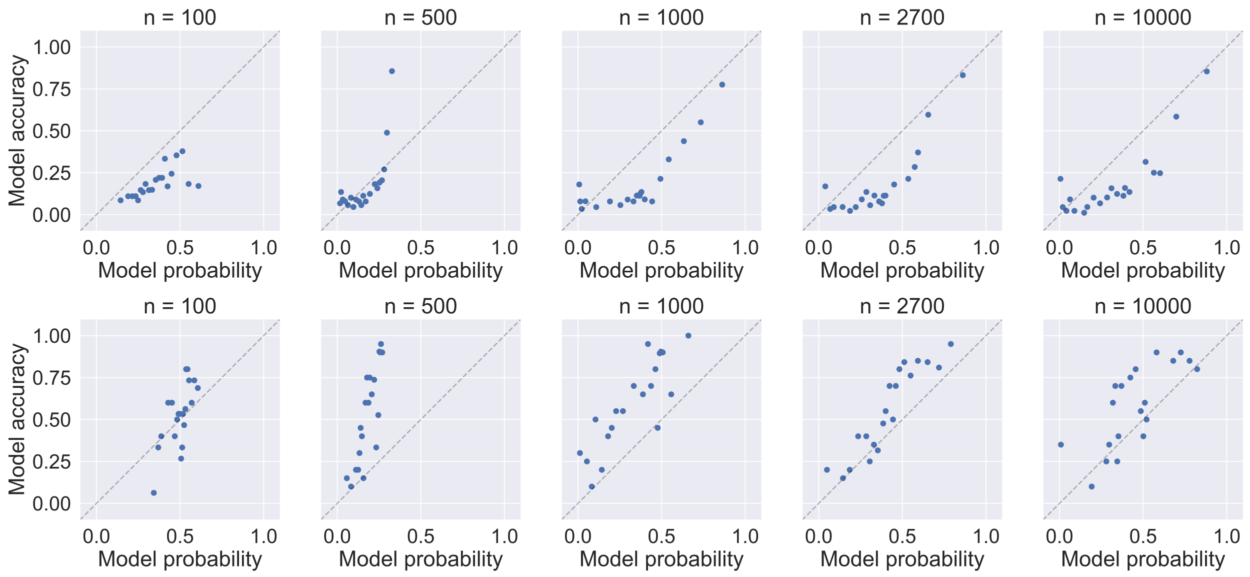


Figure 10: **Calibration curves by number of training examples.** We train the model to produce verbalized probabilities (numbers) on the Add-subtract training set. Curves show calibration performance for the Multiply-divide (top) and Multi-answer (bottom) evaluation sets using Expected Value decoding over output tokens (rather than greedy decoding). Beyond around $n = 2700$, continuing to train does not improve generalization.

C.2 Comparing results using greedy and EV uncertainties

By verbally expressing uncertainty using a number (e.g. “Confidence: 84%”), models can cover a wide range of probability values even if greedy decoding is used. In comparison, expressing uncertainty using words limits models to five categories in our setup, corresponding to the discrete confidence scores [10%, 30%, 50%, 70%, 90%]. Taking an expected value (EV) over output tokens allows models to give intermediate scores (e.g. 0.5× “High” (70%) + 0.5× “Medium” (50%) = 60% confidence). The difference between greedy and EV uncertainties is more pronounced when the number of finetuning or k-shot examples is low.

Table 4: **Performance of finetuned models using greedy and EV uncertainties.**

Setup	**Multi-answer**		**Multiply-divide**	
	MSE	**MAD**	**MSE**	**MAD**
Verbalized numbers (greedy)	22.0	16.4	15.5	19.0
Verbalized numbers (EV)	**21.5**	**14.6**	**15.0**	**18.9**
Verbalized words (greedy)	29.0	24.0	**12.7**	**10.6**
Verbalized words (EV)	**26.0**	**21.7**	12.7	13.3

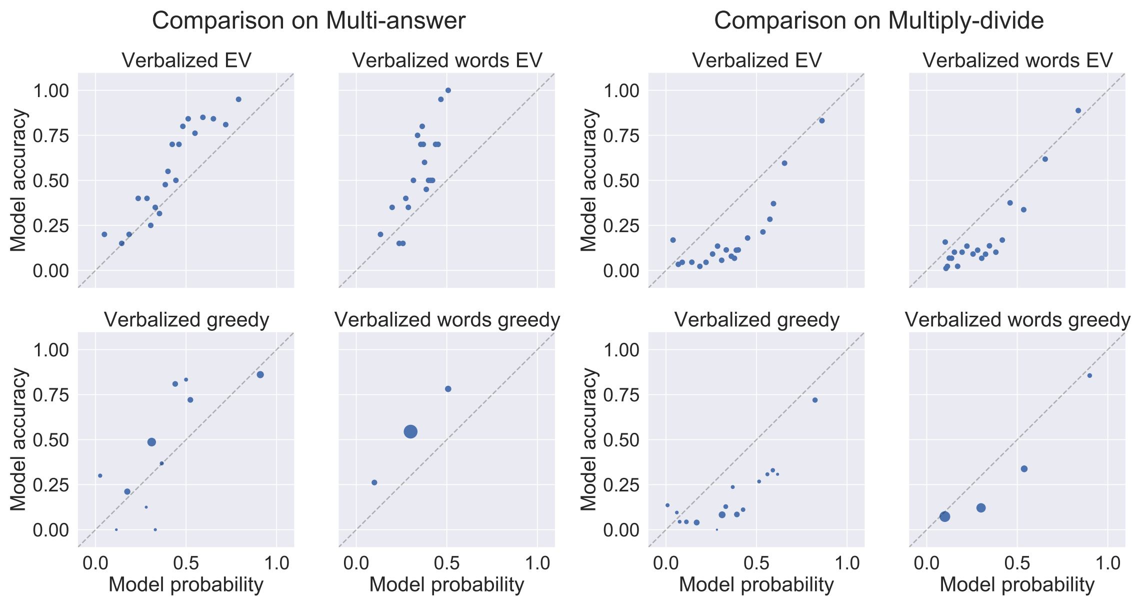


Figure 11: **Calibration curves using greedy and EV uncertainties.**

C.3 Changing the training set from Add-subtract to Multiply-divide

Table 5: **Calibration performance of models with a different training set.** In contrast to the results in the main text (where models are trained on Add-subtract), here we train models on the Multiply-divide set and we evaluate on both Add-subtract and Multi-answer. We find that calibration on the Multi-answer evaluation set is worse than when training on Add-subtract. One reason is that there is a bigger shift in the "label distribution" from training to evaluation. GPT-3's answers are less accurate on Multiply-divide and so probabilities above 50% are barely represented in the training set but make up most tasks in Multi-answer. The label distributions (i.e. distribution of accuracy for GPT-3 on the arithmetic tasks) are shown in Figure 8.

Setup	**Add-subtract**		**Multi-answer**	
	MSE	**MAD**	**MSE**	**MAD**
Verbalized numbers (finetune)	17.0	9.9	36.3	40.7
Verbalized words (finetune)	16.4	**6.8**	**30.5**	**30.2**
Answer logit (zero-shot)	**15.5**	14.3	37.4	33.7
Indirect logit (finetune)	17.3	15.0	43.9	49.9
Constant baseline	20.1	8.5	40.1	39.5

C.4 Correlations between probability types

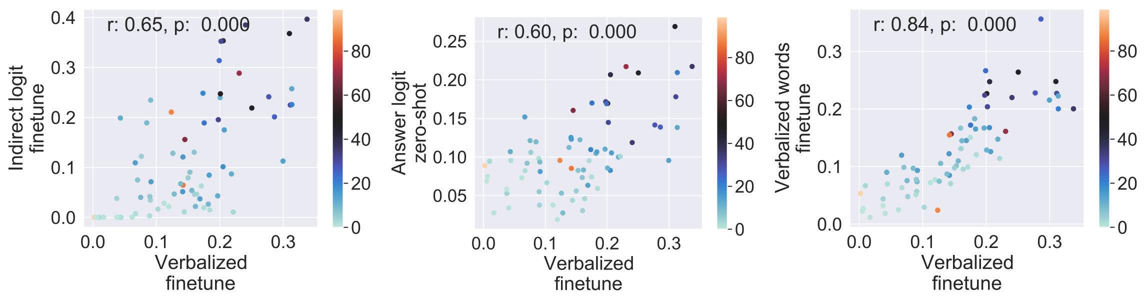


Figure 12: **Correlation between verbalized probability and logit setups.** Using the Multiply-divide evaluation set, we calculate each setup's MSE on each task and difficulty level, then plot the results. The colorbar shows GPT-3's accuracy on the arithmetic questions. While correlation between the two verbalized uncertainty types – expressing uncertainty either in numbers (e.g. 45%) or words ("Confidence: Low") – is high, correlation to the other two types is moderate. This provides more evidence that the finetuned verbalized model isn't simply reproducing the answer logit.